OS/390®

DCE
User's Guide

SC28-1586-01

OS/390®

DCE
User's Guide

SC28-1586-01

Note

Before using this information and the product it supports, read the information in Appendix A, "Notices" on page 41.

Second Edition (March 2000)

This edition applies to Version 2 Release 9 of OS/390 DCE Base Services, OS/390 DCE User Data Privacy (DES and CDMF), OS/390 DCE User Data Privacy (CDMF) (5647-A01), and to all subsequent releases and modifications until otherwise indicated in new editions. This edition replaces SC28-1586-00.

The following statements are provided by the Open Software Foundation.

Contents

Figures

Tables

About This Book

This book describes how to perform user tasks in the Distributed Computing Environment (DCE). Performing these user tasks require the use of DCE user and administrative facilities.

More detailed information about these facilities can be found in the *OS/390 DCE Administration Guide*, SC28-1584.

You should already be familiar with basic DCE concepts. If you are not, read *Distributed Computing Environment: Understanding the Concepts*, GC09-1478.

How to Use This Book

This book is divided into the following chapters:

- Chapter 1, "OS/390 DCE User Commands" on page 1 describes the OS/390 aspects of running the OS/390 DCE commands in TSO, batch and the MVS shell. "Working with DCE User Accounts" on page 4 describes how to log in to DCE, how to use OS/390-DCE Single Sign-on, how users are authenticated, and how to display registry information.
- Chapter 2, "Using the DCE Directory Service" on page 19 describes the concepts of the DCE Directory service and how to display CDSS name space information.
- Chapter 3, "Working with Access Control Lists" on page 27 describes some of the mechanisms by which authorization is controlled and explains how to change access to DCE objects.

Any reference to DCE in this book specifically means DCE for the MVS/ESA™ operating system, unless otherwise noted.

Conventions Used in This Book

This book uses the following typographic conventions:

Bold	**Bold** words or characters represent system elements that you must enter into the system literally, such as commands, options, or path names.
Italic	*Italic* words or characters represent values for variables that you must supply.
`Example font`	Examples and information displayed by the system appear in `constant width type style`.
[ ]	Brackets enclose optional items in format and syntax descriptions.
{ }	Braces enclose a list from which you must choose an item in format and syntax descriptions.
|	A vertical bar separates items in a list of choices.
< >	Angle brackets enclose the name of a key on the keyboard.
...	Horizontal ellipsis points indicate that you can repeat the preceding item one or more times.
\	A backslash is used as a continuation character when entering commands from the shell that exceed one line (255 characters). If the command exceeds one line, use the backslash character **** as the last nonblank

character on the line to be continued, and continue the command on the next line.

This book uses the following keying conventions:

<**Alt-***c*> The notation <**Alt-***c*> followed by the name of a key indicates a control character sequence.

<**Return**> The notation <**Return**> refers to the key on your keyboard that is labeled with the word Return or Enter, or with a left arrow.

Entering commands When instructed to enter a command, type the command name and then press <**Return**>.

Product Name

The product name **OS/390 DCE** refers to the DCE services on MVS/ESA.

Where to Find More Information

Where necessary, this book references information in other books, using shortened versions of the book title. For complete titles and order numbers of the books for all products that are part of OS/390, see the *OS/390 Information Roadmap*, GC28-1727. For complete titles and order numbers of the books for OS/390 DCE, refer to the publications listed in the “Bibliography” on page 55.

Online Books

All the books belonging to the OS/390 DCE library are available as online publications. They are included in the *IBM OS/390 Collection*, SK2T-6700.

All the books in the Online Library are viewable, without change, on these IBM operating platforms: OS/390, VM, OS/2®, DOS, and AIX/6000®. The same book can be viewed on any of these platforms using the IBM BookManager® Library Readers™ for OS/2, Windows, and DOS, or any of the IBM BookManager READ licensed programs for OS/390, VM, OS/2, Windows, DOS, or AIX/6000.

The booklet included with the Online Library provides details on accessing the OS/390 DCE online publications.

How to Send Your Comments

Your feedback is important in helping to provide the most accurate and high-quality information. If you have any comments about this book or any other OS/390 documentation:

- Visit the home page at: `http://www.ibm.com/s390/os390`
- Fill out one of the forms at the back of the book and return it by mail, by fax, or by giving it to an IBM representative.

Summary of Changes

Summary of Changes
for SC28-1586-01
OS/390 Version 2 Release 9

This book contains information previously presented in *OS/390 DCE: OpenEdition® DCE User's Guide*, SC28-1586-00, which supports OS/390 Release 1.

The following summarizes the changes to that information.

- As part of the name change of OpenEdition® to OS/390 UNIX System Services, occurrences of OpenEdition have been changed to OS/390 UNIX System Services or its abbreviated name, OS/390 UNIX. OpenEdition may continue to appear in messages, panel text, and other code locations.

This book includes terminology, maintenance, and editorial changes that are not marked. Technical changes or additions to the text and illustrations, however, are indicated by a vertical line to the left of the change.

Chapter 1. OS/390 DCE User Commands

In OS/390 DCE, the DCE user commands can be run in any of the OS/390 environments: TSO, batch, or MVS shell. This chapter describes how these commands are run in each environment.

OS/390 DCE User Command Names

In OS/390 DCE, there is a slight difference in some user command names when running from TSO (or batch) and running from OMVS shell.

While OS/390 DCE command names can be entered in either uppercase or lowercase in TSO (or batch), these commands can be entered only in lowercase in the MVS shell. Also, some OS/390 DCE commands in the shell contain underscores. All OS/390 DCE commands in TSO are entered without underscores.

Table 1 lists the commands described in this book and the corresponding command names for running them in TSO, batch, and the shell.

In batch, these names correspond to the PROC names that are shipped with the OS/390 DCE product for these facilities.

Table 1. OS/390 DCE Commands

Facility	TSO and Batch	OMVS Shell
ACL Editor	ACLEDIT	acl_edit
CDS Control Program	CDSCP	cdscp
DCE Control Program	DCECP	dcecp
DCE Login	DCELOGIN	dce_login
Destroy Login Context	KDESTROY	kdestroy
DTS Control Program	DTSCP	dtscp
List Kerberos tickets	KLIST	klist
Refresh Credentials Cache	KINIT	kinit
Registry Editor	RGYEDIT	rgy_edit
Save DCE password in RACF® DCE segment	STOREPW	storepw

For simplicity, most of the examples in this book are in command line or interactive modes only. That is, commands are *entered* from TSO or from the shell.

Entering Arguments to OS/390 DCE Commands

Do not enter user commands that use all uppercase or mixed-cased arguments from the ISPF command line. Aside from the parameters that these commands require, the term *argument* can also refer to subcommands of the DCE, RPC, CDS, and DTS control programs and the Registry Editor. Arguments that are all uppercase or are of mixed-case are converted to all lowercase characters.

This creates a problem for arguments (such as CDS directory names) that have mixed or all uppercase characters. For example, if you enter the following command:

```
tso cdscp show dir /.:/TestDir
```

from the ISPF command line, the control program looks for the directory /.:/testdir, and returns an error message.

If the command has all uppercase or mixed-cased arguments, enter it from the OMVS shell or from native TSO only.

In the case of DCECP, RPCCP, CDSCP, DTSCP, DCELOGIN, and the Registry Editor, you can also run the command without any arguments from the ISPF command line to start an interactive session. When in the interactive session, you can enter all uppercase, all lowercase, or mixed-case arguments to these commands.

Running the User Commands in Batch

This section describes the possible ways that you can start the control program in batch.

| The names of the PROCS to run these commands that are shipped with the OS/390 DCE product are listed in the second column of Table 1 on page 1. For example, to run the DCECP server command:

```
server show/.:/hosts/cellname/config/srvrconf/testsrvr
```

The following JCL passes the control program commands and arguments in the parameter (PARM) field:

```
//* JCL TO EXECUTE THE DCECP SERVER SHOW COMMAND
//JOB1  JOB...
//GO       EXEC   PROC=DCECP,
//         PARM='/-c server show/.:/hosts/cellname/config/srvrconf/testsrvr'
```

The following is an example of a control program command that is run using an inline dataset. DCECP processes the command as though it was started interactively.

```
//* JCL TO EXECUTE THE DCECP SERVER SHOW COMMAND
//JOB1  JOB...
//GO       EXEC   PROC=DCECP
  .
  .
//SYSIN    DD *
server show/.:/hosts/cellname/config/srvrconf/testsrvr
/*
```

In the following example, the JCL refers to a dataset name that contains the DCECP command and arguments. DCECP processes the command as though it was started interactively.

```
//* JCL TO EXECUTE THE DCECP SERVER SHOW COMMAND
//*
//JOB1  JOB...
//GO       EXEC   PROC=DCECP
.
.
//SYSIN    DD     DSN ='DCECP.INPUT',DISP=SHR
```

In this example, the dataset DCECP.INPUT contains the following statement:

```
server show/.:/hosts/cellname/config/srvrconf/testsrvr
```

You can also have a SYSIN DD statement that points to a file that contains the control program subcommands and the required arguments:

```
//* JCL TO EXECUTE THE CDSCP LIST OBJECT COMMAND
//*
//JOB1  JOB...
//GO       EXEC   PROC=CDSCP
.
.
//SYSIN    DD     DSN ='CDSCP.SOURCE.FILE',DISP=SHR
```

In this example, the dataset CDSCP.SOURCE.FILE contains the following line:

```
list object /.:/eng/*
```

Notes:

1. Control program subcommands and arguments must be in the local code page when they are passed by any of the following:
   - A SYSIN DD statement pointing to a file
   - A SYSIN DD statement pointing to an inline dataset
   - The parameter (PARM) field

   This is because the control program command processes the subcommands as though they were run interactively.
2. When you are creating JCL, be sure that there are no sequence numbers in columns 72-80.

Entering Commands that Cannot Fit in One Line

When entering user commands interactively, if the command exceeds one line (255 characters), to continue to the next line use the backslash character (\) as the last non-blank character at the end of the current line.

Command Input and Redirecting Output

Like any MVS program, the OS/390 DCE user commands can get input from, or redirect output to, a file—both HFS (hierarchical file system) and PDS (partitioned data set).

File redirection is discussed in more detail in the *OS/390 DCE Administration Guide*.

Echoing Input Commands to the Standard Output File

You can set the **_EUV_ECHO_STDIN** environment variable to **1** to display the invocation of a user command in the standard output file.

This is especially useful when running the commands in batch, where the output file gives you an indication of which commands failed, if any failure should occur.

Setting environment variables is discussed in the *OS/390 DCE Administration Guide*.

Working with DCE User Accounts

As a DCE user, you have an account in the Registry database. You can access this account using your DCE user ID (also known as principal) and password. This section describes the tasks that you can do with your DCE account.

Logging In to DCE

You can log into DCE interactively from either TSO or the shell, or you can log in from batch. When you log into DCE, you log into an account. You supply your name as identified in the account and the correct password. The password is used in the authentication of your account, as described in “Authentication of User Identities” on page 11.

Note: If the clocks on the Security Server host system and the user machine are not synchronized to within 5 minutes of each other, you may receive a password validation error, and you cannot log into DCE. See your system administrator for help.

If you have multiple DCE user IDs, you can switch among these DCE identities. This is described in “Using Concurrent Multiple Identities” on page 9.

Logging In to DCE Interactively

You can log into DCE interactively from either TSO or from the shell.

To log in from TSO, use the **dcelogin** utility:

```
dcelogin
principal_name
password
```

To log in from the shell, use the **dce_login** utility:

dce_login *principal_name password*

If you enter **dcelogin** or **dce_login** with no arguments, you are prompted for the principal name and password.

Depending on how the DCE administrator created your DCE account, your DCE *principal_name* can be a single name such as **john**, or a multi-part name such as **finance/john** (to denote that user John belongs to the Finance department).

If you belong to multiple cells in a multi-cell environment, qualify the *principal_name* with the cell name. For example, if you want to log in as user **finance/john** in the cell **XYZCompany**, use the following fully qualified name as your DCE user ID:

/.../XYZCompany/finance/john

For information about cell names, see the *OS/390 DCE Administration Guide*.

You can use the **-c** option of **dce_login** to validate and certify the login context. If the **-c** option is supplied, the command certifies the principal's identity. This option is used only as a means to enhance the performance of the application program. To use this option, you must be running with the POSIX user ID (uid) of the **root** user application program.

Hiding DCE User Passwords: Logging in to DCE from TSO is not encouraged because the password is not hidden from view when you enter it.

If you log into DCE from the MVS shell, you can hide the password by using the **hide** option of the OMVS subcommand facility. For example:

1. Enter **dce_login** from the shell prompt.
2. When prompted, enter your DCE user ID.
3. Go to the **OMVS Subcommand** mode by pressing the appropriate PF key.
4. Enter the **hide** command.
5. Enter your password. It is hidden from view.

Logging In to DCE in Batch

In batch mode, the user ID and password are passed as parameters in the JCL. Figure 1 shows a portion of a sample JCL.

```
//JOB1      JOB ...
//STEP1     EXEC DCELOGIN,PARM='principal password'
//STEP2     EXEC PGM=PROG1
```

Figure 1. Example JCL for Logging In to DCE

dcelogin and the UNIX System Services su Command

If you use the **su** command to switch to a different OS/390 user ID, use care before performing a **dcelogin**. If the _EUV_SEC_KRB5CCNAME_FILE environment variable was not set to point to a file in the new OS/390 user's home directory first, you might:

- Overwrite the **krb5ccname** file in the home directory of the user that you are issuing the **su** command from (if you have **rw** permission to the file)

  or

- Create a **krb5ccname** file in the original OS/390 user's home directory that does not have read or write permission (if the file does not exist and you have **rw** authority to the user's home directory)

If you do not have permission to the **krb5ccname** file, an attempt to log into DCE fails. Also, if you do not perform a **dcelogin** on the new OS/390 user ID, you inherit the DCE login of the original OS/390 user ID, provided you have the necessary permissions to the **krb5ccname** file.

Use the following as an example to prevent this:

1. **/home/joeuser/>** dce_login joeuser joepwd
2. **/home/joeuser/>** su janeuser
3. **/home/joeuser/>** export _EUV_SEC_KRB5CCNAME_FILE=/home/janeuser/krb5ccname
4. **/home/joeuser/>** dce_login janeuser janepwd

The preceding path name prompt is only for clarification of what happens during **su** processing for the home directory. Enter **exit** and press Enter to return to the OS/390 user ID **joeuser**. You are returned to the **joeuser** session with no changes to the original DCE identity information.

OS/390 DCE Single Sign-on Authentication

OS/390 DCE single sign-on logs an authenticated OS/390 user into DCE. DCE single sign-on support is started automatically when a DCE application starts. If the user is not logged into DCE, OS/390 DCE single sign-on attempts to log the user into DCE. Before the user starts the DCE application, the RACF administrator must set up the user for single sign-on processing. (A functionally-equivalent external security manager may be used in place of RACF.)

To request OS/390 DCE single sign-on support, be sure that:

- A RACF DCE segment is created for you
- The AUTOLOGIN flag in your OS/390 user ID's RACF DCE Segment is set to YES for single sign-on processing. The default AUTOLOGIN setting is NO.
- Your current OS/390 DCE password is saved in the RACF database, using the **storepw** command. For subsequent password changes, using the **-r** option on the **storepw** command changes the password in both the DCE registry and the RACF database at the same time.
- You have not set _EUV_AUTOLOG equal to NO in your environment variables file

DCE single sign-on does not provide support for servers that must log into DCE. Servers in OS/390 DCE still need to keep their DCE passwords in a keytab file and save the old password when the DCE password gets changed, until it is no longer needed.

OS/390 DCE single sign-on supports only a single-user environment.

Note: If you are enrolled for DCE single sign-on and have stored an incorrect password, you get error messages when you start any DCE application, including DCE commands. These messages occur until you store the correct DCE password in your RACF DCE segment, set the **_EUV_AUTOLOG** environment variable to **No**, or explicitly login to DCE.

Hiding DCE User Passwords with storepw

The **storepw** command requires that you enter your new DCE password twice. Using the command from TSO is not recommended because the password is not hidden from view when you enter it. If you use **storepw** in the OS/390 shell, you can hide the password using the **hide** option of the OMVS subcommand facility. To hide the password, do these steps:

1. Enter:

   ```
   storepw
   ```

2. When the prompt for the password appears, go to OMVS subcommand mode by pressing the appropriate function key.
3. Enter:

   ```
   hide
   ```

4. Enter your password.
5. Repeat steps 2 and 3.

The _EUV_AUTOLOG Environment Variable

You can use the _EUV_AUTOLOG environment variable to disable single sign-on processing, if the administrator enabled it in your DCE segment. The only value it can have is NO, so _EUV_AUTOLOG=NO is the only valid setting. Any other setting is ignored. This variable must be declared in the environment variable file (commonly called the **envar** file) in your home directory for it to be shared in the TSO and batch environments.

Single Sign-on and the UNIX System Services su Command

If you use the **su** command to switch from one OS/390 user ID to another and if you are enrolled in OS/390 DCE single sign-on, you must take certain steps before starting a DCE application under the new identity. You must first set the _EUV_SEC_KRB5CCNAME_FILE environment variable to point to a file in the new user ID's home directory before starting a DCE application. If you do not do this, one or more of these errors may occur:

- You may overwrite the **krb5ccname** file in the home directory of the user ID from which you are issuing the **su** command (if you have **rw** permission to the file)
- You may create a **krb5ccname** file in the home directory of the original user ID that the original OS/390 user does not have permission to read or write to (if the file does not exist and you have **rw** authority to the user ID's home directory)
- You may start a DCE application that fails (if you do not have **rw** permission to the **krb5ccname** file of the original user ID)
- You may start a DCE application that uses the DCE identity that the previous OS/390 user ID was logged on as (if you have **rw** permission to the **krb5ccname** file of the original user ID)

Furthermore, if you do not have permission to the **krb5ccname** file, DCE single sign-on fails. Also, if the new OS/390 user ID is not enrolled in single sign-on, the user must set the _EUV_SEC_KRB5CCNAME_FILE environment variable and perform an explicit **dcelogin**.

The following example shows that both OS/390 user IDs are enrolled in DCE single sign-on. The DCE command **klist** causes an automatic sign-on.

```
/home/joeuser/> klist
/home/joeuser/> su janeuser
/home/joeuser/> export _EUV_SEC_KRB5CCNAME_FILE=/home/janeuser/krb5ccname
/home/joeuser/> klist
```

The path name prompt in this example is only to clarify what happens during **su** processing regarding the home directory. Note that to return to the original user ID **joeuser**, you simply enter **exit** and press Enter. You return to the **joeuser** session with any original DCE identity information unchanged.

Why the Ticket Cache File is Important

Each time you log into DCE, a ticket cache file (also known as the **credentials cache file**) is created for the principal name that was used when you logged in. This file is the physical manifestation of your login context. The login context represents your DCE identity and its associated privileges. The ticket cache file is an HFS file and is stored in the **/opt/dcelocal/var/security/creds** directory.

The KRB5CCNAME Variable

The KRB5CCNAME variable specifies the HFS path name of the user's ticket cache file. That is, the value of this variable determines the login context that is currently in effect. All ticket cache files are in the **/opt/dcelocal/var/security/creds** directory. An example setting of this variable is as follows:

```
KRB5CCNAME=FILE:/opt/dcelocal/var/security/creds/JOHN.CACHE.DT940624.TM194139
```

The Credentials Cache Name File

At the same time that the ticket cache file is created when you log in to DCE, an HFS file called the Credentials Cache Name file is updated. The Credentials Cache Name file has only one line, the declaration of the KRB5CCNAME variable. The KRB5CCNAME variable points to the user's ticket cache file. The Credentials Cache Name file, in effect, points to the user's ticket cache file and thus determines the user's effective login context.

When you log into DCE, the value of the KRB5CCNAME variable is updated to the HFS pathname of your ticket cache file.

Note: You can log into DCE in various ways: running the **DCELOGIN** command from TSO or batch, running **dce_login** from the shell, or calling the **sec_login** APIs from an application program. Remember that in all cases, you are updating the Credentials Cache Name file.

By default, the Credentials Cache Name file is created in your home directory with the name **krb5ccname**. For example, if John's home directory is **/home/john**, his default Credentials Cache Name file is **/home/john/krb5ccname**.

You can specify the Credentials Cache Name file to create with a different path name by setting another environment variable. (This is discussed in “The _EUV_SEC_KRB5CCNAME_FILE Environment Variable.”)

The Credentials Cache Name file can be either an OS/390 dataset or an HFS file.

The Credentials Cache Name file plays a key role in maintaining multiple concurrent DCE identities, where there are multiple login contexts (and therefore, multiple ticket cache files). This is discussed later in this section.

Note: If two or more users share the same home directory, they share the same Credentials Cache Name file. Thus, they overwrite each other's login context each time one of them logs in to DCE. In this case, you can specify a different Credentials Cache Name file for each user, by setting the _EUV_SEC_KRB5CCNAME_FILE environment variable. This is discussed in the next section. (However, it is not good practice to share home directories.)

The _EUV_SEC_KRB5CCNAME_FILE Environment Variable

The _EUV_SEC_KRB5CCNAME_FILE variable specifies the path name of the Credentials Cache Name file. The default value of this environment variable is **$HOME/krb5ccname**, that is, a file named **krb5ccname**, in your home directory.

Because you can set its value to any path name, the _EUV_SEC_KRB5CCNAME_FILE environment variable is useful when a user performs multiple DCE logins, where multiple Credentials Cache Name files can be created, (that is, one Credentials Cache Name file for each DCE identity). By setting the value of this variable to the appropriate Credentials Cache Name file, you can specify the effective DCE identity of the user.

This variable is primarily used in non-shell environments (TSO or batch).

Note: This variable must be declared in the **envar** file in your home directory, for it to be shared in the TSO and batch environments.

Figure 2 shows the relationship between the _EUV_SEC_KRB5CCNAME_FILE environment variable, the Credentials Cache Name file, and the ticket cache.

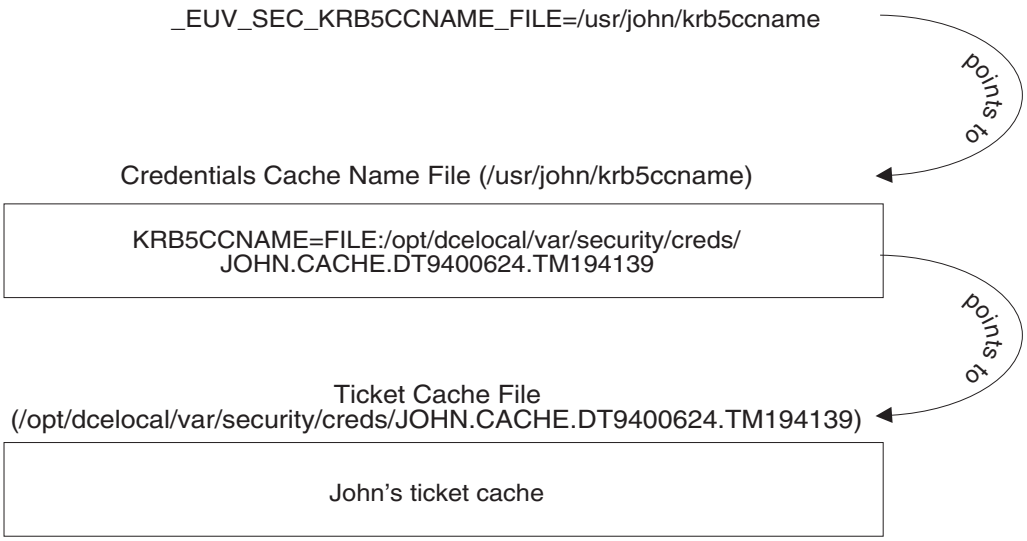


Figure 2. _EUV_SEC_KRB5CCNAME_FILE, Credentials Cache Name file, and the ticket cache

Using Concurrent Multiple Identities

If you have multiple DCE accounts, you can perform multiple logins to DCE, then use the different DCE identities to do different tasks. If you log in multiple times using different DCE identities, you can make DCE create a different ticket cache file for each of the identities that you used to log into DCE. You can then use each of these ticket cache files to switch among the multiple identities that you used to log into DCE. You can accomplish this by pointing to the appropriate Credentials Cache Name file before performing each DCE Login.

For the TSO and batch environments, there are two alternative ways of referring to a Credentials Cache Name file:

- Define the DDNAME EUVSKRB5 and point it to the appropriate Credentials Cache Name file.
- Set the value of the _EUV_SEC_KRB5CCNAME_FILE environment variable (in your **envar** file) to the appropriate Credentials Cache Name file.

In the shell, exporting the _EUV_SEC_KRB5CCNAME_FILE environment variable defines the appropriate Credentials Cache Name file.

The following sections provide examples that illustrate each of these methods.

Referring to the Credentials Cache Name File in TSO

The following example shows how to use the ALLOC command to refer to an MVS data set as the Credentials Cache Name file:

```
ALLOC FI(EUVSKRB5) DS('TS99999.TICKET.NAME.USER2')SHR REUSE
```

The following example shows how to refer to an HFS file as the Credentials Cache Name file:

```
ALLOC FI(EUVSKRB5) PATH('/home/john/john_ccf') PATHOPTS(ORDWR) PATHDISP(KEEP,KEEP)
```

Referring to the Credentials Cache Name File in Batch

The following example shows how to refer to an MVS data set as the Credentials Cache Name file in the JCL:

```
//EUVSKRB5 DD DSN='TS99999.TICKET.NAME.USER2',DISP=SHR
```

The following example shows how to refer to an HFS file as the Credentials Cache Name file in the JCL:

```
//EUVSKRB5 DD PATH='/home/john/john_ccf'
```

Using _EUV_SEC_KRB5CCNAME_FILE in TSO or Batch

The **envar** file can be referenced from TSO or batch as the default source of environment variable declarations. As a result, if the _EUV_SEC_KRB5CCNAME_FILE environment variable is declared in the **envar** file, the Credentials Cache Name file that is pointed to by the value of this variable becomes the basis of the effective login context for the TSO session or batch job.

Setting environment variables in batch and TSO is discussed in the *OS/390 DCE Administration Guide*.

Referring to the Credentials Cache Name File from the Shell

From the shell, use the **export** command to specify the Credentials Cache name file that you want in the _EUV_SEC_KRB5CCNAME_FILE environment variable.

For example, to inherit the login context of user john:

```
export _EUV_SEC_KRB5CCNAME_FILE=/home/john/john_ccf
```

Using EUVSKRB5 with Multiple Identities

The following example illustrates the use of the Credentials Cache Name file to inherit and to switch between two login contexts in the three environments: TSO, batch, and the shell.

In this example, user Tom has two DCE identities: **tomA** and **tomB**.

Tom logs in to DCE as **tomA** and wants to submit a batch job under **tomA**'s identity. Tom performs the following:

1. From TSO, Tom logs into DCE as **tomA**.

   When Tom performs a DCE Login as user **tomA**, the Credentials Cache Name file of DCE user **tomA** is created (by default) in his home directory, for example, **/home/tom/krb5ccname**, and the login context of user **tomA** is established.

2. To submit a batch job that runs under the login context of **tomA**, Tom does not have to perform any additional step because he is currently referring to **tomA**'s login context.

Now, Tom wants to log into DCE as **tomB** and submit a batch job under **tomB**'s DCE identity. Tom performs the following:

1. From TSO, he runs the ALLOC command to allocate and refer to **tomB**'s Credentials Cache Name file:

   ```
   ALLOC FI(EUVSKRB5) PATH='/home/tom/tomB'
   ```

   If Tom simply logs into DCE as **tomB**, he overwrites the Credentials Cache Name file of **tomA** (**/home/tom/krb5ccname**). That is, the default Credentials Cache Name file is updated to point to **tomB**'s ticket cache file. By running the ALLOC command, Tom is directing **tomB**'s Credentials Cache Name file to a different file than **tomA**'s Credentials Cache Name file.

2. Tom logs into DCE as **tomB**.

   When Tom logs into DCE as **tomB**, his Credentials Cache Name file is created as **/home/tom/tomB**.

3. To submit a batch job under :hp2/tomB's DCE identity, Tom has to refer to **tomB**'s Credentials Cache Name file in the JCL as follows:

   ```
   //EUVSKRB5 DD DSN='/home/tom/tomB'
   ```

Finally, Tom wants to perform some tasks in the shell under the DCE identity of **tomB**. Note that the _EUV_SEC_KRB5CCNAME_FILE environment variable is still set to the default value (**/home/tom/krb5ccname**), which points to **tomA**'s login context. If Tom wants to perform some tasks in the shell as **tomB**, he has to refer to **tomB**'s Credentials Cache Name file. He can do this by resetting the value of the _EUV_SEC_KRB5CCNAME_FILE environment variable to **tomB**'s Credentials Cache Name file (using the **export** command) as follows:

```
export _EUV_SEC_KRB5CCNAME_FILE=/home/tom/tomB
```

Authentication of User Identities

When you log into a DCE account, the Security Service checks your password against the password stored for you in the registry database. If the password is the same as the one in the registry database, the Security Service obtains your ticket-granting ticket. The ticket-granting ticket is evidence that the Security Service accepts you as an authenticated user and will provide you with your privilege attributes.

Privilege attributes consist of your principal name and the groups to which you belong. When you request access to objects, privilege attributes determine your permissions to those objects.

Privilege attributes that the Security Service provides are certified. Privilege attributes that sources other than the Security Service provide are known as uncertified privilege attributes. Other network services accept certified privileges but might not accept uncertified privilege attributes. Depending on whether your privilege attributes are certified or not, the kinds of access that you are allowed to DCE objects can differ.

Note: Unless stated otherwise, the term privilege attributes refers to certified privilege attributes only.

Ticket-Granting Tickets and Service Tickets

Your ticket-granting ticket permits you to request and receive tickets to DCE services. The tickets that enable your access to DCE services are called **service tickets**. Your service ticket indicates to a server that you are an authenticated user. If your service ticket states that either you or a group that you belong to is authorized to use the resources of that server, the server accepts your request. If you find that you require a service that you cannot access, see your system administrator for help.

Ticket-granting tickets and service tickets have lifetimes. Your system administrator and the policies of your installation determine the lifetime of a ticket. If your ticket-granting ticket expires, you are no longer an authenticated user. Your access to objects other than those on the local machine is stopped, and your ability to use DCE services is limited. To prevent this from happening, you use the **klist** command to find out the lifetime of your ticket-granting ticket, then reauthenticate by running the **kinit** command before your ticket-granting ticket expires. If your system administrator configured your account as able to renew service tickets, the Security Service renews them automatically.

Note: The lifetime of a service ticket can never exceed the time remaining on your ticket-granting ticket.

Using the kinit Command to Reauthenticate

To run **kinit**, enter:

kinit [*principal_name*]

The *principal_name* is the login name of the principal whose tickets are to be re-authenticated. It must be the same principal name under which you are currently logged in.

The **kinit** command prompts for the password associated with *principal_name*. If you enter it correctly, your ticket-granting ticket is renewed. If you do not enter your password correctly, **kinit** displays an error message. If you receive this message, run **kinit** again and enter the correct password. You must run **kinit** before your ticket-granting ticket expires.

Note: The **kinit** command has other options not described here. You can use these options to specify an alternative ticket cache and request forwardable, proxiable, and renewable tickets. See the *OS/390 DCE Command Reference* for more information about this command.

Using klist to Display Your Privilege Attributes and Tickets

To display your tickets and your privilege attributes, use the **klist** command. In OS/390 DCE, this command lists:

- Your OS/390 user ID
- Your DCE principal name
- Groups in which you are a member
- Tickets to services the Security Service component granted you
- The date and time your ticket-granting ticket expires

To run **klist** to display your current tickets, enter it with no options. The **klist** command shows your privilege attributes, expiration information, and service ticket information. (You can use the **klist -e** option to view current and expired tickets.)

Privilege Attributes: The first part of the **klist** output, shown in Figure 3, lists your privilege attributes. It shows:

1 Your local OS/390 user ID

2 Your fully qualified principal name, or global principal

3 The UUIDs and names of your cell

4 Your principal name (without the cell name and DCE global identifier)

5 All the groups in which you are a member

```
Local OS Identity Information:
       User:  JOHN  1
DCE Identity Information:
       Global Principal: /.../dresden.com/music/mozart  2
       Cell:        5ad96550-80c4-11ca-b26c-08001e039431 /...dresden.com  3
       Principal:  00000066-80c5-11ca-b600-08001e039431 music/mozart  4
       Group:       00000003-80c4-11ca-b201-08001e039431 composers  5
       Local Groups:
              00000003-80c4-11ca-b201-08001e039431 composers
```

Figure 3. Example: klist Display — Privilege Attributes

Expiration Dates and Times: The second part of the **klist** display, shown in Figure 4, lists the dates and time that your ticket-granting ticket, account, and password expire:

1 The date and time your ticket-granting ticket expires. Before this happens, re-initialize it by running **kinit** or logging in again to DCE.

2 The date and time your account expires. If your account has expired, you are not able to log into DCE. To remedy this, your system administrator must change the account expiration date in the registry.

3 The date your password expires. When this happens, you must enter a new password before you can log into DCE.

```
Identity Info Expires: 94/10/03:12:07:18  1
Account Expires:       94/12/31:12:00:00  2
Passwd Expires:        94/10/31:12:00:00  3
```

Figure 4. Example: klist Display — Expiration Dates and Times

Tickets: The final part of the **klist** display, shown in Figure 5 on page 14, lists your ticket information and the name of your ticket cache:

1 The tickets labeled `Server` are the tickets used after you logged in to obtain your privilege attributes. The display for all principals has these entries.

2 The tickets labeled `Client` show your ticket-granting ticket and your service tickets. In the listing for each ticket after the word `Client` is the name of the privilege server, which grants your privilege attributes after the Security Service authenticates your identity. The name of the server to which you have tickets is shown after the `Server` entry, and the next line shows the dates and times when these tickets are valid.

For example, in Figure 5 on page 14, the last line shows that the principal has a ticket to the server called **file_server**. The lifetime of this ticket is from

1:24 and 2 seconds p.m. on 10/2/94 to 12:07 and 18 seconds p.m. on 10/3/94. (The time is shown in 24-hour format.)

```
Kerberos Ticket Information:
Ticket cache:  /tmp/JOHN.CACHE.DT941002.TM120645
Default principal: music/mahler
Server: krbtgt/dresden@dresden.com 1
   valid 94/10/02:12:07:18 to 94/10/03:12:07:18
Server:dce/rgy@dresden.com
   valid 94/10/02:12:07:20 to 94/10/03:12:07:18
Server:dce/ptgt@dresden.com
   valid 94/10/02:12:07:49 to 94/10/03:12:07:18
Client:dce/ptgt@dresden    Server:krbtgt/dresden@dresden.com 2
   valid 94/10/02:12:07:50 to 94/10/03:12:07:18
Client:dce/ptgt@dresden.com   Server:dce/rgy@dresden.com
   valid 94/10/02:12:07:53 to 94/10/03:12:07:18
Client:dce/ptgt@dresden.com   Server:file_server@dresden.com
   valid 94/10/02:13:24:02 to 94/10/03:12:07:18
```

Figure 5. Example: klist Display — Tickets

Using kdestroy to Destroy Your Tickets

Use the **kdestroy** command to invalidate the tickets you have acquired. In DCE, there is no concept of physically logging out (that is, entering a logout command). Rather, the concept of logging out can be thought of in terms of destroying all your unexpired tickets. To run **kdestroy**, enter it from either TSO or the shell with no options.

Using the Registry Editor

This section describes the use of the Registry Editor to change your password and display registry information.

All instructions in this section describe how to use the Registry Editor in interactive mode, although you can also use the Registry Editor in command line mode. For a complete description on the use of the Registry Editor, see the *OS/390 DCE Administration Guide*.

Running and Exiting from the Registry Editor

To run the Registry Editor from TSO, enter:

rgyedit

To run the Registry Editor from the shell, enter:

rgy_edit

The following prompt is displayed:

```
rgy_edit=>
```

To exit from the Registry Editor, enter:

```
rgy_edit=> quit
```

Changing Your Password

If you have the required DCE permissions, you can use the Registry Editor to change your DCE password.

Note: To be able to change your password, you must have the following DCE permissions:

- Read (r) and Update (u) permissions on the account's principal.
- Read (r) permission on the registry Policy object.

The following example shows how to change your password by using the Registry Editor.

1. At the `rgy_edit=>` prompt, enter **change**.

   You are prompted for your principal name.

   ```
   Change Account> Enter account id [pname]:
   ```

2. Enter your principal name.

   You are prompted for the name of the group associated with your account.

   ```
   Enter account group [gname]:
   ```

3. Enter the group name.

   You are prompted for the name of the organization associated with your account.

   ```
   Enter account organization [oname]:
   ```

4. Enter the organization name.
5. Press Enter at the next three prompts to leave the information unchanged:

   ```
   Enter new account id [pname]: (mahler)
   Enter new account group [gname]: (symphonists)
   Enter new account org [oname]: (classic)
   ```

   You are prompted to change your password.

   ```
   Change password? [y/n]? (n)
   ```

6. Enter **y**.

   You are prompted for a new password.

   ```
   Enter new password:
   ```

7. Enter your new password.

   You are prompted to retype your new password.

   ```
   Retype new password:
   ```

8. Enter your new password again.

   You are prompted for your current password to ensure that you are authorized to change it.

   ```
   Enter your password:
   ```

9. Enter your current password.
10. Press Enter at the following prompts to leave the information unchanged:

```
Enter new misc info: ()
Enter new home directory: (/)
Enter new shell: (/bin/csh)
Password valid [y/n]? (y)
Enter new expiration date [yy/mm/dd or 'none']: (none)
Allow account to be server principal [y/n] (y)
Allow account to be client principal [y/n] (y)
Account valid for login [y/n]? (y)
Allow account to obtain post-dated certificates [y/n]?    (n)
Allow account to obtain forwardable certificates [y/n]   ? (y)
Allow certificates to this account to be issued
        via TGT authentication [y/n]? (y)
Allow account to obtain renewable certificates [y/n] (y)
Allow account to obtain proxiable certificates [y/n] (n)
Allow account to obtain duplicate session keys [y/n] (n)
Good since date [yy/mm/dd]: (1994/05/15.12:04)
Create/Change auth policy for this acct [y/n] (n)
```

After you press Enter at the `Change Account> Enter account id [pname]:` prompt, you see the `rgy_edit=>` prompt.

11. Enter **quit** to end the Registry Editor session.

Note: If you are enrolled for single sign-on in your RACF DCE segment and you have not disabled this function with _EUV_AUTOLOG=NO in your **envar** file, be sure to use the OS/390 DCE **storepw** command to change your password in the DCE segment (and, optionally, in the DCE registry).

Displaying Registry Information

If your system administrator sets the appropriate permissions, you can view account information in the Registry database. This section briefly describes how to use the Registry Editor to display accounts, principals, groups, and organizations.

Displaying Accounts: You can use the **view** subcommand (abbreviated as **v**) to display account information:

1. Enter **rgyedit** in TSO or **rgy_edit** in the shell.

   You see the following prompt:

   ```
   rgy_edit=>
   ```

   By default, you are placed in the account domain.

2. Enter the **view** command and the name of the principal whose account you want to display. (If you do not enter a principal name, all accounts in the cell are displayed.) Figure 6 shows the account for the principal **mahler**.

```
rgy_edit=>v mahler
mahler[symphonists classic]:6XPbd13hiftzTE:24583:12::/fugue/mahler::
rgy_edit=>
```

Figure 6. Account for the Principal mahler

If you enter the **view** subcommand with the **-f** option, you can display the account's administrative information as shown in Figure 7 on page 17.

```
rgy_edit=> v mahler -f
mahler[symphonists classic]:6XPbd13hiftzTE:24583:12::/fugue/mahler::
created by: /...//dresden.com/root 1994/06/11.19:57
Changed by: /.../dresden.com/root 1994/07/09.19:57
 password is: valid, was last changed: 1994/07/09.00:41
 Account expiration date: 1994/11/01.10:00
 Account MAY be a server principal
 Account MAY be a client principal
 Account is: valid
 Account CAN NOT get post-dated certificates
 Account CAN get forwardable certificates
 Certificates to this service account MAY be issued via TGT authentication
 Account CAN get renewable certificates
 Account CAN NOT get proxiable certificates
 Account CAN NOT have duplicate session keys
 Good since date: 1994/06/11.19:57
 Max certificate lifetime: 8h
 Max renewable lifetime: 1w
rgy_edit=>
```

Figure 7. Example: Displaying Accounts

Displaying Groups: You can use the **view** subcommand (abbreviated as **v**) to display group information by performing the following steps:

1. At the rgy_edit=> prompt, change to the group domain of the Registry database:

   ```
   rgy_edit=> do group
   ```

2. Enter the **view** subcommand and the name of the group that you want display. (If you do not enter a group name, all groups are displayed.) The following example displays the group **symphonists**. The display includes the group name and UNIX number.

   ```
   rgy_edit=> v symphonists
   symphonists 193
   ```

   To display members of a group, enter the **view** subcommand with the **-m** option. The following example displays members in the group **symphonists**:

   ```
   rgy_edit=> v symphonists -m
   symphonists 193
     3 members: brahms, britten, mahler
   ```

Displaying Organizations: You can use the **view** subcommand (abbreviated to **v**) to display organization information by performing the following steps:

1. At the rgy_edit=> prompt, change to the organization domain of the Registry database:

   ```
   rgy_edit=> do org
   ```

2. Enter the **view** subcommand and the name of the organization that you want to display. (If you do not enter an organization name, all organizations are displayed.) The following example displays the organization **classic**. The display includes the organization name and UNIX number.

   ```
   rgy_edit=> v classic
   classic 12
   ```

3. To display members of an organization, enter the **view** subcommand with the **-m** option.

Displaying Principals: You can use the **view** subcommand (abbreviated as **v**) to display principal information by performing the following steps:

1. At the `rgy_edit=>` prompt, change to the principal domain of the Registry database:

   ```
   rgy_edit=> do principal
   ```

2. Enter the **view** subcommand and the name of the principal that you want to display. (If you do not enter a principal name, the Registry Editor displays all principals.) The following example displays the principal **mahler**. The display includes the principal name and UNIX number.

   ```
   rgy_edit=> v mahler
   mahler  24583
   ```

   To display full information about the principal, use the **-f** option:

   ```
   rgy_edit=> v mahler -f
   mahler                                        24583
     Uuid:        0000013e-1bbd-2e6e-9400-10005ac92e21
     Primary:   pr    Reserved:  --
     Quota:  unlimited
   ```

   To display all the groups in which the principal is a member, enter the **view** subcommand with the **-m** option. The following example displays the groups in which **mahler** is a member:

   ```
   rgy_edit=> v mahler -m
   mahler 24583
    Member of 1 group:
    symphonists
   ```

Chapter 2. Using the DCE Directory Service

This chapter provides the concepts of the DCE Directory Service, then describes how to use the CDS Control Program to view the CDS namespace.

DCE Directory Service Concepts

The DCE Directory Service resembles a telephone directory that provides a phone number when given the name of a person. When you provide the DCE Directory Service with the unique name of a person, server, or resource, it returns the network address and other information associated with that name. The Directory Service stores addresses and other relevant information as attributes of the name.

Attributes can contain the name of an organizational division, such as European Sales, a location, such as the first floor of Building A, or a telephone number. You can search for a name by supplying one or more of its attributes. For example, by using the search criteria of **John Smith** and **Chicago**, you can use the Directory Service to produce a list of telephone numbers for users in Chicago named John Smith. Search capabilities are limited to the global part of the Directory Service environment.

You typically use directory services indirectly, through an application interface. When you create a name for a resource and refer to it by that name, your application can interact with the Directory Service for you. The following examples show some of the ways you can use the Directory Service:

- When you log into a system, you enter a name and password. The Directory Service helps the login program locate an authentication server, which verifies your identity in an authentication database.
- You enter the name of a computer conference or electronic bulletin board. The Directory Service provides an address, which allows you to connect to the conference service.
- You enter a report in a problem-tracking database. Although the database was recently moved to a new node, you are not aware of the change because the problem-tracking application you use refers to it by name only. The Directory Service stores the current network address and provides it to the problem-tracking application and to any other application that requests it.

The names that the DCE Directory Service works with exist in a cell environment. Depending on the location of the cell in which a name exists, different components of the DCE Directory Service work with the name. The *OS/390 DCE Administration Guide* describes the different components of the DCE Directory Service that may be present in the cell. In addition, the location of the cell affects the name itself. The *OS/390 DCE Administration Guide* also describes naming conventions.

CDS Clearinghouse

CDS directories can be replicated. Each physical copy of a directory is called a replica. Replicas are stored in structures called clearinghouses. You can think of a clearinghouse as a collection of directory replicas on one CDS server.

Viewing the Structure and Contents of the CDS Namespace

This section describes how to use the CDS control program to display the structure and the contents of the CDS namespace. In particular, you can use the **show** and **list** commands of the CDS control program to display this information.

Starting the CDS Control Program

You can start the CDS control program (CDSCP) by entering the following command from TSO or the shell:

cdscp

This brings up an interactive session and the following prompt displays, at which you can enter CDSCP subcommands:

```
cdscp>
```

You can also use the CDS control program in command line mode, wherein the **cdscp** command is entered with its subcommands on the command line, for example:

cdscp show clerk

And you can run the **cdscp** subcommands in batch, as "Running the User Commands in Batch" on page 2 describes.

Using the show Command

You can use the **show** command to display the names of all the directories stored in the clearinghouse. You can also display the current values of any or all attributes associated with any name in the name space.

The basic syntax of all **show** commands is as follows:

show *entity-type entity-name*

The *entity-type* is the type of CDS entity for which you want to display information and *entity-name* is a complete directory specification ending with a simple name (the full CDS name of the entity). In this form, the **show** command displays the current values of all attributes associated with the entity that you specify.

You cannot use wildcard characters in the directory specification of the entity-name, but you can use them in the terminating (rightmost) simple name.

You can also display the name of an attribute or attributes:

show *entity-type entity-name attribute-name*

The *attribute-name* is the name of a particular attribute associated with the entity that you specify. To display the values of two or more attributes in one command, separate the attribute names with single spaces on the command line.

show Command Examples

To display the current values of all attributes associated with the local clerk, use the **show clerk** command. Figure 8 shows an example.

```
cdscp> show clerk

                        SHOW
                       CLERK
                          AT   1994-10-15-15:56:50
                Creation Time = 1994-10-09-17:03:32.32
     Authentication failures = 0
             Read Operations = 1068
                  Cache Hits = 137
              Cache bypasses = 433
            Write operations = 1250
    Miscellaneous operations = 590
```

Figure 8. Showing Attributes of the Local Clerk

To display all of the object entries stored in the **/.:/sales** directory, use the **show object** command. Figure 9 shows an example.

```
cdscp> show object /.:/sales/*

                      SHOW
                    OBJECT  /.../abc.com/sales/stats_disk
                        AT   1994-11-09-15:41:07
              CDS_CTS = 1994-10-15-13:09:47.000000003/08-00-2b-1c-8f-1f
              CDS_UTS = 1994-10-15-13:09:47.000000003/08-00-2b-1c-8f-1f
            CDS_Class = class1
     CDS_ClassVersion = 1.0

                      SHOW
                    OBJECT  /.../abc.com/sales/region01
                        AT   1994-11-09-15:41:07
              CDS_CTS = 1994-10-15-13:09:47.000000003/08-00-2b-1c-8f-1f
              CDS_UTS = 1994-10-17-08:59:50.000000006/08-00-2b-1c-8f-1f
            CDS_Class = class1
     CDS_ClassVersion = 1.0
```

Figure 9. Showing Object Entries in a Directory

To display all clearinghouse object entries stored in the root directory, use the **show object** command. Figure 10 on page 22 shows an example.

```
cdscp> show object /.:/* with CDS_Class = CDS_Clearinghouse

                        SHOW
                      OBJECT  /.../Paris_CH
                          AT   1994-10-18-15:53:07
                     CDS_CTS =
1994-10-15-13:09:47.000000003/08-00-2b-1c-8f-1f
                     CDS_UTS =
1994-10-17-08:59:50.000000006/08-00-2b-1c-8f-1f
                   CDS_Class = CDS_Clearinghouse
            CDS_ClassVersion = 1.0

                        SHOW
                      OBJECT  /.../NY1_CH
                          AT   1994-10-18-15:54:06
                     CDS_CTS =
1994-09-11-11:10:51.000000003/08-00-2b-1c-8f-1f
                     CDS_UTS =
1994-09-11-12:59:50.000000006/08-00-2b-1c-8f-1f
                   CDS_Class = CDS_Clearinghouse
            CDS_ClassVersion = 1.0
```

Figure 10. Showing Clearinghouse Object Entries in the Root Directory

To display all soft links stored in the **/.:/mfg** directory that contain the string **robot**, use the **show link** command. Figure 11 shows an example.

```
cdscp> show link /.:/mfg/*robot*

                    SHOW
                  SOFTLINK  /.../abc.com/mfg/robot01
                        AT   1994-10-15-15:54:40
               CDS_CTS = 1994-08-15-13:09:47.000000003/08-00-2b-1c-8f-1f
               CDS_UTS = 1994-10-13-08:59:50.000000006/08-00-2b-1c-8f-1f
        CDS_LinkTarget = /.../abc/mfg/robotics_controller1

                    SHOW
                  SOFTLINK  /.../abc.com/mfg/new_robot
                        AT   1994-10-15-15:54:41
               CDS_CTS = 1994-09-11-13:09:47.000000003/08-00-2b-1c-8f-1f
               CDS_UTS = 1994-10-13-08:59:50.000000006/08-00-2b-1c-8f-1f
        CDS_LinkTarget = /.../abc/mfg/robotics_controller2
       CDS_LinkTimeout = :
               Expiration = 1994-10-15-00:00:00.0
                Extension = 0
```

Figure 11. Showing Soft Links in a Directory

Permissions Required to Use the show Command

To use the **show** command you must have read permission to the name you want to display. To specify some entity types, you require additional permissions. Table 2 shows the permissions that you require.

Table 2 (Page 1 of 2). show Commands and Required Permissions

Command	Required Permissions
show cached clearinghouse	Read permission to the clerk.
show cdscp confidence	No specific permissions are required.
show cdscp preferred clearinghouse	No specific permissions are required.

Table 2 (Page 2 of 2). show Commands and Required Permissions	
Command	**Required Permissions**
show cached server	Read permission to the clerk.
show cell	Read permission to the cell root directory.
show child	Read permission to the child pointer. If you specify a wildcard child name, you also need read permission to the parent directory.
show clearinghouse	Read permission to the clearinghouse. If you specify a wildcard clearinghouse name, you also need read permission to the cell root directory.
show clerk	Read permission to the clerk.
show directory	Read permission to the directory. If you specify a wildcard directory name, you also need read permission to the directory's parent directory.
show link	Read permission to the soft link. If you specify a wildcard soft link name, you also need read permission to the directory that stores the soft link.
show object	Read permission to the object entry. If you specify a wildcard object entry name, you also need read permission to the directory that stores the object entry.
show replica	Read permission to the directory of which the replica is a member.
show server	Read permission to the server. **Note:** This command is not available in OS/390 DCE.

For more information about the **show** command, see the *OS/390 DCE Administration Guide*.

Using the list Command

You can use the **list** command to display names that match a name you specify, or to display a list of the object entries, soft links, or child pointers in a directory.

The basic syntax of all **list** commands is as follows:

list *entity-type entity-name*

The *entity-type* is the type of entity that you are listing, and the *entity-name* is a complete directory specification ending with a simple name.

list Command Examples

To display the names of all the object entries stored in the **/.:/eng** directory, use the **list object** command. Figure 12 shows an example.

```
cdscp> list object /.:/eng/*
                          LIST
                        OBJECT  /.../abc.com/eng/*
                            AT   1994-10-15-15:53:06

sales_stats
test_stats
triton
work_disk1
work_disk2
```

Figure 12. Listing Object Entries in a Directory

To display all names that begin with the letter **t** for object entries stored in the **/.:/eng** directory, use the **list object** command. Figure 13 on page 24 shows an example.

```
cdscp> list object /.:/eng/t*

                        LIST
                      OBJECT  /.../abc.com/eng/t*
                         AT   1994-10-15-15:53:06
test_stats
triton
```

Figure 13. Listing Object Entries That begin with t in a Directory

To display the names of all directories that have a **CDS_Convergence** attribute set to a value of **medium** and are one level below the root, use the **list directory** command. Figure 14 shows an example.

```
cdscp> list directory /.:/* with CDS_Convergence = medium

                        LIST
                   DIRECTORY  /.../abc.com/*
                          AT  1994-10-15-15:53:06
eng
mfg
sales
```

Figure 14. Listing Directories with a Medium Convergence

Permissions Required to Use the list Command

To use the **list** command you must have read permission to the name you want to display. To specify some entity types, you require additional permissions when you use the **with** *attribute-name* = *attribute-value* clause. Table 3 shows the permissions that you require.

Table 3. list Commands and Required Permissions

Command	Required Permissions
list child	Read permission to the directory that stores the child pointer. If you use a **with** *attribute = attribute-value* clause in the command, you also need read or test permission to the selected child pointers.
list clearinghouse	Read permission to the directory that stores the associated clearinghouse object entry. If you use a **with** *attribute = attribute-value* clause in the command, you also need read or test permission to the selected clearinghouses.
list directory	Read permission to the parent directory. If you use a **with** *attribute = attribute-value* clause in the command, you also need read or test permission to the selected directories.
list link	Read permission to the directory that stores the soft link. If you use a **with** *attribute = attribute-value* clause in the command, you also need read or test permission to the selected soft links.
list object	Read permission to the directory that stores the object entry. If you use a **with** *attribute = attribute-value* clause in the command, you also need read or test permission to the selected object entries.

For more information about the **list** command, see the *OS/390 DCE Administration Guide*.

Filtering the Output of show and list Commands

You can use the **with** *attribute-name = attribute-value* clause to limit the action of a **show** or **list** command to only those entities with the attribute value that you specify. You cannot use the **with** *attribute-name = attribute-value* clause with **show** or **list** commands that you enter to display information about clerks or servers.

Chapter 3. Working with Access Control Lists

An access control list (ACL) is an authorization mechanism that assigns permissions that control access to DCE objects. When you log in, permissions are conferred on the principal on which you are logged in. When you try to access a resource, your permissions are checked against the permissions listed in the ACL for that resource.

ACLs protect the following DCE objects.

- Principals, groups of principals, and organizations that the DCE Security Service manages
- Files and file system directories that the Distributed File Service manages
- Distributed Time Service (DTS) servers
- CDS directories and entries
- CDS clients and servers that have ACLs restricting the use of their management operations (for example, creating a clearinghouse)
- GDS entries that the GDS's own ACL mechanism manages

An ACL consists of multiple ACL entries that define:

- Who can use an object
- What operations can be performed on the object.

You can modify an ACL to allow or prevent access to objects that you own by using the ACL Editor.

For more information about authorization and how it works, refer to "Authentication of User Identities" on page 11 and to the *OS/390 DCE Administration Guide*.

Access Control List Interpretation

Part of the information associated with a user account is the principal name and the group (or groups) associated with the principal name. The universal unique identifiers (UUIDs) that represent the principal's name and group names are known as the privilege attributes of the principal.

If the Authentication Service supplied the privilege attributes for a principal, they are known as certified privilege attributes. Principals without certified privilege attributes are allowed only unauthenticated access to objects. Unauthenticated access, if it is allowed at all, is usually more restrictive than authenticated access.

When a principal requests access to a DCE object associated with an ACL, the ACL Manager for the object reads the list of ACL entries. The ACL Manager grants the access permissions specified in the first ACL entry that matches any of the privilege attributes supplied for the principal. When the ACL Manager finds a match, it stops checking the entries.

Privilege Attributes Inherited by Processes

Processes that a principal creates or spawns inherit the privilege attributes of the principal. For example, if you log in and are authenticated, any application you start inherits your authenticated privilege attributes. Processes that the application spawns inherit your privilege attributes and pass them down to the processes that they start.

Some servers are written to run as separate authenticated principals. For these servers, your system administrator creates an account in the Registry database. When you start these servers, the server process performs the equivalent of a user login, receives its privilege attributes, and runs under its own identity, not yours.

Access Control List Entries and Masks

ACL entries are of the following form:

type[**:***key*] **:***permissions*

The following example ACL entry, shown in Figure 15, sets permissions for a principal in the local cell, named **bach**. The ACL entry type is **user**, the key that identifies the specific principal is **bach**, and the permissions are **rwxid**. Colons separate the entry components.

Note: Not all types of ACL entries require you to enter a key.

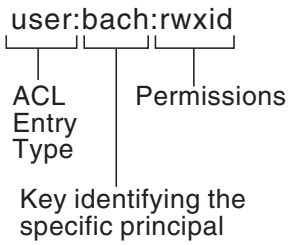


Figure 15. Sample ACL Entry

ACL entry types define entries for:

- Principals and groups:
  - Principals and groups in the local cell
  - Principals and groups in foreign cells
  - All principals in the local and all foreign cells for whom individual ACL entries were created
  - All principals in the local and all foreign cells whose privilege attributes do not match any of the other ACL entries
- Masks used for authenticated and unauthenticated users. Refer to the *OS/390 DCE Administration Guide* for more information about masks.
- Entry types (to be defined in the future) that can be copied and displayed (if not interpreted) by dissimilar DCE Extended releases. Refer to the *OS/390 DCE Administration Guide* for more information about entry types.

Access Control List Entry Types for Principals and Groups

If a principal or group is not authenticated, the unauthenticated mask further constrains the permissions in the entry. The **mask_obj** mask further constrains all entries for authenticated principals except **user_obj** and **other_obj** entries.

Table 4 shows the entry types for principals and groups and the purpose and entry format for each. The table uses the following syntax variables:

- The *principal_name* is the name the principal uses to log in.
- The *group_name* is the name of the group defined in a Registry database.
- The *cell* is the global path name of the foreign cell in the format */.../name*.
- The *permissions* are the permissions the ACL Manager for the object makes available.

Table 4 (Page 1 of 2). ACL Entry Types for Principals and Groups

ACL Entry Type	Purpose	ACL Entry Format
user_obj	Establishes permissions for the real or effective user of the object. This type is similar to the UNIX owner entry.	user_obj:*permissions*
group_obj	Establishes permissions for members of the real or effective group of the object. This type is similar to the UNIX group entry.	group_obj:*permissions*
other_obj	Establishes permissions for all others in the local cell, unless they: • Are specifically named in ACLs of entry type **user** • Are members of a group named in an ACL with an entry type of **group** • Match the principal the **user_obj** or **group_obj** entry indicates. This type is similar to the UNIX other entry.	other_obj:*permissions*
user	Establishes permissions for a specific principal in the local cell. You must identify the principal by supplying a principal name in the ACL entry.	user:*principal_name*:*permissions*
group	Establishes permissions for members of a specific group in the local cell. You must identify the group by supplying a group name as a key.	group:*group_name*:*permissions*
foreign_user	Establishes permissions for a specific principal in a foreign cell. You must identify the principal by supplying a principal name and cell name as a key.	foreign_user:*cell_name/principal_name*:*permissions*
foreign_group	Establishes permissions for a specific group in a foreign cell. You must identify the group by supplying a group name and a cell name as a key.	foreign_group:*cell_name/group_name*:*permissions*

Table 4 (Page 2 of 2). ACL Entry Types for Principals and Groups		
ACL Entry Type	**Purpose**	**ACL Entry Format**
foreign_other	Establishes permissions for other principals in a specific foreign cell that are not specifically named in ACL entries of entry type **foreign_user** or are members of a group named in an ACL entry of type **foreign_group**. You must identify the foreign cell by supplying a cell name as a key.	foreign_other:*cell_name*:*permissions*
any_other	Establishes permissions for all other principals in local or foreign cells unless they match a more specific entry in the ACL.	any_other:*permissions*
unauthenticated	Establishes the permissions set that masks the permission set in a privilege attribute entry that corresponds to a principal whose privilege attributes were not certified by an authority such as Privilege Service.	unauthenticated:*permissions*

Group Permissions and Project Lists

Principals gain group permissions from their project list, which lists all the groups in which a principal or alias is a member. The principals's access rights to an object come from the logical OR of permissions granted to every group with an entry in the ACL in which the principal is a member.

Note: The principal obtains rights only from the name or alias with which the principal logs in, not from both names and aliases.

For example, suppose an ACL contains the following entries:

```
user_obj:crwxid-
group_obj:crwx---
other_obj:-r-----
group:composers:crwx---
user:bach:crwx---
user:mozart:crwx---
group:performers:--w-idt
```

Assume that user **cole** is a member of the group **composers** and the group **performers**. Because **cole** obtains permissions from all groups, his access permissions are **crwxidt**.

The DCE Security Service provides a method to prevent a group from being included in a project list, and thus preventing the group's permissions from being accrued as part of the project list.

Denying Access

When you create an ACL entry for a principal or group, you grant only the permissions you specify in the ACL entry. To deny a principal all access to an object, create an ACL entry that contains a dash in place of the permissions. For example to deny all access to user **mozart**, use the entry:

user:mozart:-

If you want to deny access to a specific principal or group, select the most specific entry type available. Generally for principals, the most specific entry type is **user** or **foreign_user**; for groups, the most specific entry type is **group** or **foreign_group**.

Note: If the principal is the object's owner or a member of the object's group, you must use the **user_obj** or **group_obj** entry types to ensure that access is denied.

Using the ACL Editor

The ACL Editor creates, modifies, and displays ACL entries.

Using this command, you can:

- Create and modify ACL entries for DCE objects in the local cell and foreign cells
- Display the permissions an object's ACL Manager implemented for the object
- Create and modify the masks that restrict allowable permissions. For more information on masks, see the *OS/390 DCE Administration Guide*.

This section gives general instructions for using the ACL Editor. It describes how to start the command and how to use the help facility. For more information on the ACL Editor invocation options and subcommands, see the *OS/390 DCE Command Reference*.

Starting the ACL Editor

You can run the ACL Editor in interactive mode, where it prompts you for the information it needs, or in command-line mode, where you enter all the information that the ACL Editor needs on the command line. In command-line mode, you can perform only one ACL Editor operation at a time. In DCE, the ACL Editor commands can also be run in batch mode by submitting a JCL or in interactive mode by the file's processing a CLIST.

Note: When you start the ACL Editor for the ACL of an object, that ACL is not locked: multiple users can edit the ACL at the same time. If this happens, each change can overwrite previous changes. To avoid this problem, only one user should be assigned permission to change a particular ACL. If this is not possible, the **change** authority to an ACL should be granted to as few users as possible.

Starting the ACL Editor in Interactive Mode: To start the ACL Editor in interactive mode, enter:

acledit *pathname* (TSO)

or

acl_edit *pathname* (from the shell)

Where:

pathname Specifies the object for which you want to display or modify the ACL. If the object is in another cell, you must enter the fully qualified path name.

For example, to edit the object named **opus** in the file server named **my_filesystem**, use a path name such as:

/.../dresden.com/my_filesystem/opus

The ACL Editor displays the following prompt:

```
sec_acl_edit>
```

Starting the ACL Editor in Command-Line Mode: To start the ACL Editor in command-line mode, enter:

acledit *pathname command_line_subcommand* [*acl_entry*] (TSO)

or

acl_edit *pathname command_line_subcommand* [*acl_entry*] (from the shell)

Where:

pathname	Specifies the object whose ACLs are to be displayed or modified. If the object is in another cell, you must enter the fully qualified path name. For example, to edit the object named **opus** in the file server named **my_filesystem**, enter: **/.../dresden.com/my_filesystem/opus**
command_line_subcommand	Specifies the ACL Editor subcommand to run.
acl_entry	Specifies the acl entry in the form: *type***[:***key* **]:***permissions*

For example, to set **rwx** permissions for user **bach** to the DCE object named **opus** in the local cell, enter:

acledit opus -m user:bach:rwx (TSO)

or

acl_edit opus -m user:bach:rwx (from the shell)

In JCL, the PARMS field is limited to 100 characters. If the ACL Editor parameters exceed 100 characters, use the **-stdin** option and supply the parameters in the SYSIN of the JCL. For example:

```
ACLEDIT   EXEC PGM=ACLEDIT
             PARM=('-stdin')
             .
             .
             .
//SYSIN      DD *
             .
             .
```

Starting the ACL Editor on Directory Service Leaf Objects: By default, the ACL Editor fully resolves the path name you enter when you start the command. Sometimes, a path name resolves to a leaf object in the Directory Service and to an object in some other DCE component that supports ACLs, such as the Registry. In these situations, you must use the **-e** option to edit the leaf object in the Directory Service.

For example, a print server that implements an ACL Manager that limits access to printing services may have a top-level path name (that is, catalog point or name to which the object exports its location bindings) in the following form:

/.../dresden.com/print_server

The component **print_server** may be both a leaf in the CDS namespace and the top level directory in the print server namespace.

To edit the ACL associated with the top level of the print server, start the ACL Editor as follows:

acledit /.../dresden.com/print_server (TSO)

or

acl_edit /.../dresden.com/print_server (from the shell)

To edit the ACL associated with the leaf object in the Directory Service, start the ACL Editor with the **-e** option as follows:

acledit -e /.../dresden.com/print_server (TSO)

or

acl_edit -e /.../dresden.com/print_server (from the shell)

Saving Changes During an ACL Editor Session

When you make changes using the ACL Editor in command-line mode, they are saved. If you are using the ACL Editor in interactive mode, you may want to save your changes and continue the session. To save them, enter:

```
sec_acl_edit> commit
```

Exiting from the ACL Editor

To end the ACL Editor session and save the changes you specified in the session, use the following **exit** command:

```
sec_acl_edit> exit
```

To end the ACL Editor session without saving the changes you specified, use the following **abort** command:

```
sec_acl_edit> abort
```

Using the ACL Editor Help Facility

In interactive mode, the ACL Editor **help** subcommand displays help information. If you enter **h** or **?**, the ACL Editor shows a list of all subcommands and available topics. Figure 16 shows an example of the help display.

```
sec_acl_edit> h
Known commands are:
ab[ort]         as[sign_file]   co[mmit]        d[elete]
e[xit]          g[et_access]    h[elp]          k[ill_entries]
l[ist]          m[odify]        p[ermissions]   ce[ll]
sec_acl_entry   su[bstitute]    t[est_access]   ?
```

Figure 16. Example: Using the ACL Editor Help Facility

The sec_acl_entry topic displays help information for ACL entries.

If you enter **h** and a subcommand name, the ACL Editor shows information about the command. Figure 17 shows an example of the help display.

```
sec_acl_edit> h as
assign_file --
  Assign the sec_acl entries contained in the specified file
  to the object
  Usage:
  as[sign] FILENAME
```

Figure 17. Example: Displaying the ACL Editor Help Information

Specifying Names in ACL Entries

The ACL Editor uses a default cell name for the principals and groups specified in **user_obj**, **group_obj**, **other_obj**, **user**, and **group** ACL entry types. The default cell name identifies the cell in which the principal or group is registered. When you create a user or group ACL entry for a principal or group in the local cell, you do not have to enter the full path name, but only the principal or group name. To complete the path name, the ACL Editor uses the default cell name for that particular ACL.

The default cell name is usually the name of the local cell. The primary use of the default cell is to allow you to copy ACLs to a cell other than the one in which they were created. You can use the ACL Editor **cell** subcommand to name the default cell. The default you set remains in place until you change it with another **cell** subcommand.

You can use the **list** subcommand to display the default cell name.

The **user_obj**, **group_obj**, and **other_obj** entries do not require a principal or group name as a key. The ACL Manager can determine to whom the entries apply. However, the specific principals and groups derived from those entries are assumed to exist in the default cell.

When you create ACLs with an entry type of **foreign_user**, **foreign_group**, or **foreign_other**, you must specify the cell in which the principals exist. For **foreign_user** and **foreign_group** entries, use the fully qualified name of the principal or group in the following form:

cell_name/principal_name

The *cell_name* consists of the DCE global prefix (**/...**) followed by a slash, and then the name of the cell. For example, for the principal named **bach** at the cell named **dresden.com**, you enter:

/.../dresden.com/bach

For an entry type of **foreign_other**, you need to specify only the cell name. For example:

/.../dresden.com

Displaying Access Control List Entries

The ACL Editor **p[ermissions]** and **l[ist]** subcommands display the permissions available for an object and all entries in the ACL for that object.

Displaying Permissions

The exact permissions you enter for an ACL entry depend on the permissions the ACL Manager implemented for the object. The ACL Manager provides the permissions for the object and the meanings of those permissions to the ACL Editor. The ACL Editor can display the permissions appropriate for the object whose ACL you are editing, and it can discriminate between permissions that are valid and not valid for the object. To display the permission tokens for an object and their meanings, use the following ACL Editor **p[ermissions]** subcommand in interactive mode.

```
sec_acl_edit> permissions
```

The permissions you see when you use the **p** subcommand differ, depending on the permission set a particular ACL Manager supports. Figure 18 shows an example display of a permission set.

```
sec_acl_edit> permissions

Token        Description
r            read
w            write
x            execute
i            insert
d            delete
t            test
```

Figure 18. Example Permissions

In the examples that follow and in the next sections, only the invocation in TSO using the **acledit** command is used. Use **acl_edit** when running the ACL Editor from the shell.

To list the permissions for an object in command-line mode, use the **-p** option as shown in the following command that lists permissions for the object named **opus**:

acledit /.../dresden.com/my_filesystem/opus -p

Displaying Entries

To display the ACL entries for an object, use the following ACL Editor **l**[ist] subcommand in interactive mode:

```
sec_acl_edit> list
```

Note: To view the ACL entries for an object, you need **execute** permission on the directory in which the object resides and on all directories that lead to the object.

The **l**[ist] subcommand lists the ACL entries for the object specified by *pathname* when the ACL Editor was started. Figure 19 shows an example of the output this command produces.

```
# SEC_ACL for /.../dresden.com/music/printers
# Default cell = /.../dresden.com
mask_obj:crwx---
unauthenticated:-r-----
user_obj:crwx---
user:britten:crwx---
user:mahler:-rwx---
foreign_user:/.../ud.edu/pro/bach:crwxidt  #effective:crwx---
group_obj:-rwx---
group:dds:-rwx---
foreign_group:/.../china.com/writers/novelists:-r-x---
other_obj:-rwx---
foreign_other:/.../china.com:-rwx---
any_other:-r-----
extended:c417faf8-8e40-11c9-ace3-08001e55.a.b.c.a1.4.0a0b0c0d:.-rwx---
```

Figure 19. Example: ACL Entries for an Object

The first line of the display shows the path name of the object whose ACL you are editing. The default cell is listed next and then the ACL entries. In this example, the full permissions that are available to a principal are **crwxidt**. Permissions that are not available to a principal are indicated with a hyphen on the display.

If a mask restricts permissions explicitly granted in the ACL entry, the resulting permissions are listed to the right of the ACL labeled as **#effective**. In the display, the entry for **foreign_user: bach** is given explicit permissions of **crwxidt**, but the ACL entry is masked by the **mask_obj** mask. Because only permissions granted by the entry and by the mask are allowed, the permissions for the foreign user **bach** are **crwx**, not **crwxidt**.

To list the ACLs for an object in command-line mode, use the **-l** option as shown in the following command that lists ACLs for the object named **/my_filesystem/opus**:

acledit /.../dresden.com/my_filesystem/opus -l

Adding and Modifying ACL Entries

Each ACL can contain only one entry for each principal or group for entry types of **user**, **group**, **foreign_user**, and **foreign_group** and only one entry for all other entry types. For example, although the ACL for an object can contain multiple entries of type **user**, it can have only one **user** entry for a principal named **bach**. The ACL Editor subcommands that add and modify ACL entries enforce this restriction. The commands overwrite existing ACL entries so that multiple entries cannot occur.

You can add the permissions in an ACL entry in any order, and you can add the ACL entries in any order. Although the **list** subcommand shows hyphens in place of permissions not granted by an entry, you do not

need to supply hyphens when you create the entry. You can, however, use hyphens to deny access as described in “Denying Access” on page 31.

The three ACL Editor subcommands that add and modify ACL entries are:

- The **modify** subcommand adds or modifies a single ACL entry. If the entry you are modifying does not exist, **modify** adds it. If the entry you are modifying exists, **modify** replaces it with the new entry.
- The **assign** subcommand adds all the entries contained in the file whose name you specify. This subcommand overwrites all existing entries in the ACL with the ones in the named file; no previous entries remain.
- The **substitute** subcommand adds or modifies all entries. With this subcommand, you must specify each entry on the command line, separating entries with a space. As with the **assign** subcommand, all existing entries are overwritten with the ones specified in the command line.

Using the modify Subcommand

Assume the ACL for the file **opus** has the following ACL entries:

```
mask_obj:crwxid-
unauthenticated:-r-----
user_obj:crwxid-
user:bach:crwx---
user:mozart:-r-----
group_obj:crwx---
other_obj:-r-----
```

Then run the following **modify** subcommand:

```
sec_acl_edit> modify user:mozart:crwx
```

The **modify** subcommand changes only the entry for **user:mozart** as follows:

```
mask_obj:crwxid-
unauthenticated:-r-----
user_obj:crwxid-
user:bach:crwx---
user:mozart:crwx---
group_obj:crwx---
other_obj:-r-----
```

Using the assign Subcommand

To use the **assign** subcommand, first create a file containing the ACL entries in standard ACL entry format. In the file, each ACL entry should be on a separate line.

If you are running the ACL Editor from the shell, run the **assign** subcommand, specifying the name of the ACL file, in the format

as[sign] *filename*

For example, assume the file **std_acl** contains the following entries:

```
mask_obj:crwxid-
user_obj:crwxid-
group_obj:crwx---
other_obj:-r-----
user:lizt:crwx---
group:composers:-r-----
user:bach:crwx---
user:mozart:crwx---
```

The following **assign** subcommand adds the entries in **std_acl** to an ACL file:

```
sec_acl_edit> assign std_acl
```

The **assign** subcommand overwrites all ACL entries with the ones on the file **std_acl**.

If you are running the ACL Editor in batch, a DD statement must be defined for the file containing the ACL entries:

```
//filename  DD DSN='dataset-name'
```

The JCL contains a SYSIN statement that starts the **assign** command. For example:

```
//SYSIN DD*
  assign DD:filename
  exit
/*
```

You can also specify a PARM statement as follows:

```
//     PARM='/.:/dir/object -f DD:filename'
```

If you are running the ACL Editor from TSO (that is, from a CLIST), the CLIST must be edited to contain the following:

```
alloc dd(filename) ds(dataset-name)
```

The *filename* is the name of the file that contains the ACL entries and *dataset-name* is the name of the data set that is mapped to *filename*.

You can now start the **assign** command using the **stdin** option of the ACL Editor.

Using the substitute Subcommand

Assume you are in interactive mode and want to change all the ACL entries for a file. As an example, run the following **substitute** subcommand, abbreviated as **su**, on the ACL for the file **opus**:

```
sec_acl_edit> su user:mozart:crwx
```

The complete ACL (that is, all entries in the ACL) for **opus** is now:

```
user:mozart:crwx---
```

Adding and Modifying ACL Entries in Command-Line Mode

To add or modify ACL entries in command-line mode, use the **-m** (modify), **-f** (assign), or **-s** (substitute) options. For example:

- To use the **-m** option to add permissions for **user:mozart** to the ACL for the file **opus**, enter:

  acledit /.../dresden.com/my_filesystem/opus -m user:mozart:crwx

  You can add more than one entry at a time using the **-m** option. To do so, use a space between entries. For example, to add entries for user **mozart** and user **bach**, enter:

  acledit /.../dresden.com/my_filesystem/opus -m user:mozart:crw user:bach:r

- To use the **-f** option to replace the entries in the ACL for the file named **opus** with the entries in **std_acl**, enter:

  acledit /.../dresden.com/my_filesystem/opus -f std_acl

- To use the **-s** option to replace all entries with the one or ones specified on the command line, enter:

  acledit /.../dresden.com/my_filesystem/opus -s user:mozart:rw user:bach:crw

  This example replaces all entries with the two specified by the **-s** option.

Deleting Access Control List Entries

You can delete ACL entries by using the ACL Editor subcommands for the mode in which you operate.

Deleting Entries in Interactive Mode

You can use the **delete** and **kill_entries** subcommands when you are working in interactive mode. For example, use the following **delete** subcommand to delete the **user:mozart** entry:

```
sec_acl_edit> delete user:mozart
```

Use the following **kill_entries** subcommand to delete all entries, except the **user_obj** entry (if it exists):

```
sec_acl_edit> kill_entries
```

Deleting Entries in Command-Line Mode

To delete ACL entries in command-line mode, use the **-d** (delete) or **-k** (kill_entries) subcommands. For example:

- Use the **-d** subcommand to delete the **user:mozart** entry from the ACL for the file **opus**:

  acledit /.../dresden.com/my_filesystem/opus -d user:mozart

  Only the type (**user**) and key (**mozart**) are required to specify the entry to delete. You do not have to enter the entire entry.

- Use the **-k** subcommand to delete all entries, except the **user_obj** entry if it is present, from the ACL for the file **opus**:

  acledit /.../dresden.com/my_filesystem/opus -k

Copying Access Control Lists

To copy an ACL from one object to another in the shell, run the ACL Editor in command-line mode, specifying as *pathname* the object whose ACL you want to copy. Then use the **-l** subcommand to list the object's ACL entries and redirect the output to a file. For example, to list the ACL entries of the object (specified by *pathname*) and redirect the output to the file **/tmp/tmp_acl**, enter:

acl_edit *pathname* **-l > /tmp/tmp_acl**

Then, run **acl_edit** again in command-line mode, specifying as *pathname* the object to which you want to copy the ACL. Use the **-f** subcommand to replace the object's existing ACL entries with the entries saved in the file to which you redirected the output of the **-l** command. For example:

acl_edit *pathname* **-f /tmp/tmp_acl**

You can also create a shell script containing command-line subcommands to copy the ACL. In addition, the script expects the path name of the object from which the ACL is being copied to be the first argument and the path name of the object to which the ACL is being copied to be the second argument. Here is the script:

```
acl_edit $1 -l > /tmp/tmp_acl acl_edit $2 -f /tmp/tmp_acl
```

To copy an ACL in batch, define a SYSPRINT DD *dataset name* and redirect the output to a file. The PARM field of the JCL must contain the following line:

pathname `-f DD:` *dataset name*

Then, enter the ACL Editor again and use the **assign** subcommand to apply the ACL entries to the other object.

Appendix A. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10594-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

| Any references in this information to non-IBM Web sites are provided for convenience only and do not in
| any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part
| of the materials for this IBM product and use of those Web sites is at your own risk.

| IBM may use or distribute any of the information you supply in any way it believes appropriate without
| incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
522 South Road
Poughkeepsie, NY 12601-5400
U.S.A

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

This product contains code licensed from RSA Data Security Incorporated.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX/6000
CICS/ESA
IMS
MVS/ESA
OS/390
BookManager
IBM
IMS/ESA
UNIX System Services
RACF
CICS
IBMLink
Library Reader
OS/2

Microsoft, Windows, and Windows NT are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark in the United States and other countries licensed exclusively through The Open Group.

Other company, product, and service names may be trademarks or service marks of others.

Programming Interface Information

This *User's Guide* documents intended Programming Interfaces that allow the customer to write programs to obtain services of DCE.

Glossary

This glossary defines new technical terms and abbreviations used in the OS/390 DCE documentation. If you do not find the term you are looking for, refer to the index of the appropriate OS/390 DCE manual or view the *IBM Dictionary of Computing*, located at **http://www.ibm.com/networking/nsg/nsgmain.htm**.

This glossary includes terms and definitions from:

- *IBM Dictionary of Computing*, SC20-1699.
- *Information Technology—Portable Operating System Interface (POSIX),* from the POSIX series of standards for applications and user interfaces to open systems, copyrighted by the Institute of Electrical and Electronics Engineers (IEEE).
- *American National Standard Dictionary for Information Systems,* ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies may be purchased from the American National Standards Institute, 11 West 42nd Street, New York, New York 10036. Definitions are identified by a symbol (A) after the definition.
- *Information Technology Vocabulary,* developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1.SC1).
- *CCITT Sixth Plenary Assembly Orange Book, Terms and Definitions* and working documents published by the International Telecommunication Union, Geneva, 1978.
- Open Software Foundation (OSF).

The following abbreviations indicate terms that are related to a particular DCE service:

CDS	Cell Directory Service
CICS/ESA®	Customer Information Control System/ESA
DTS	Distributed Time Service
GDS	Global Directory Service
IMS/ESA®	Information Management System/ESA
RPC	Remote Procedure Call
Security	Security Service
Threads	Threads Service
XDS	X/OPEN Directory Service
XOM	X/OPEN Object Management

A

access control list (ACL). (1) GDS: Specifies the users with their access rights to an object. (2) Security: Data that controls access to a protected object. An ACL specifies the privilege attributes needed to access the object and the permissions that may be granted, to the protected object, to principals that possess such privilege attributes.

access right. Synonym for *permission*.

accessible. Pertaining to an object whose client possesses a valid designator or handle.

account. Data in the Registry database that allows a principal to log in. An account is a registry object that relates to a principal.

ACL. Access control list.

address. An unambiguous name, label, or number that identifies the location of a particular entity or service. See *presentation address*.

alias. Synonym for *alias name*.

alias name. (1) GDS: A name for a directory object that consists of one or more alias entries in the directory information tree (DIT). (2) Security: An optional alternate for a principal's primary name. Synonymous with *alias*. The alias shares the same UUID with the primary name.

attribute. (1) RPC: An Interface Definition Language (IDL) or attribute configuration file (ACF) that conveys information about an interface, type, field, parameter, or operation. (2) DTS: A qualifier used with DTS commands. DTS has four attribute categories: characteristics, counters, identifiers, and status. (3) XDS: Information of a particular type concerning an object and appearing in an entry that describes the object in the directory information base (DIB). It denotes the attribute's type and a sequence of one or more attribute values, each accompanied by an integer denoting the value's syntax.

attribute syntax. GDS: A definition of the set of values that an attribute may assume. Attribute syntax includes the data type, in ASN.1, and usually one or more matching rules by which values may be compared.

attribute type. (1) XDS: The component of an attribute that indicates the type of information given by that attribute. Because it is an object identifier, it is

unique among other attribute types. (2) XOM: Any of various categories into which the client dynamically groups values on the basis of their semantics. It is an integer unique only within the package.

attribute value. XDS, XOM: A particular instance of the type of information indicated by an attribute type.

authentication. In computer security, a method used to verify the identity of a principal.

Authentication Service. One of three services provided by the Security Service: it verifies principals according to a specified authentication protocol. The other Security services are the Privilege Service and the Registry Service.

authorization. (1) The determination of a principal's permissions with respect to a protected object. (2) The approval of a permission sought by a principal with respect to a protected object.

B

binding. RPC: A relationship between a client and a server involved in a remote procedure call.

binding handle. RPC: A reference to a binding. See *binding information*.

binding information. RPC: Information about one or more potential bindings, including an RPC protocol sequence, a network address, an endpoint, at least one transfer syntax, and an RPC protocol version number. See *binding*. See also *endpoint*, *network address*, *RPC protocol*, *RPC protocol sequence*, and *transfer syntax*.

broadcast. A notification sent to all members within an arbitrary grouping such as nodes in a network or threads in a process. See also *signal*.

C

cache. (1) CDS: The information that a CDS clerk stores locally to optimize name lookups. The cache contains attribute values resulting from previous lookups, as well as information about other clearinghouses and namespaces. (2) Security: Contains the credentials of a principal after the DCE login. (3) GDS: See *DUA cache*.

CCITT. Consultative Committee on International Telegraphy and Telephone

CDS. Cell Directory Service.

CDS control program (CDSCP). A command interface that CDS administrators use to control CDS servers and clerks and manage the name space and its contents. See also *manager*.

CDSCP. CDS control program.

cell. The basic unit of operation in the distributed computing environment. A cell is a group of users, systems, and resources that are grouped around a common purpose and that share common DCE services.

Cell Directory Service (CDS). A DCE component. A distributed replicated database service that stores names and attributes of resources located in a cell. CDS manages a database of information about the resources in a group of machines called a DCE cell.

cell-relative name. Synonym for *local name*.

child pointer. CDS: A pointer that connects a directory to a directory immediately below it in a name space. You do not explicitly create child pointers; CDS creates them for you when you create a new directory. CDS stores the child pointer in the directory that is the parent of the new directory.

class. A category into which objects are placed on the basis of their purpose and internal structure.

clearinghouse. CDS: A collection of directory replicas on one CDS server. A clearinghouse takes the form of a database file. It can exist only on a CDS server node; it cannot exist on a node running only CDS clerk software. Usually only one clearinghouse exists on a server node.

clearinghouse object entry. CDS: A special class of object entry that describes a clearinghouse. The clearinghouse object entry is a pointer to the network address of an actual clearinghouse. This pointer enables CDS to find a clearinghouse and use and manage its contents. A clearinghouse changes and manages its own object entry when necessary. The clearinghouse object entry has the same name as the clearinghouse it describes.

clerk. (1) DTS: A software component that synchronizes the clock for its client system by requesting time values from servers, calculating a new time from the values, and supplying the computed time to client applications. (2) CDS: A software component that receives CDS requests from a client application, ascertains an appropriate CDS server to process the requests, and returns the results of the requests to the client application.

client. A computer or process that accesses the data, services, or resources of another computer or process on the network. Contrast with *server*.

client context. RPC: The state within an RPC server generated by a set of remote procedures and maintained across a series of calls for a particular client. See *context handle*. See also *manager*.

compatible server. RPC: A server that offers the requested RPC interface and RPC object and that is accessible over a valid combination of network and transport protocols. It is supported by both the client and server RPC run times.

Consultative Committee on International Telegraphy and Telephone (CCITT). A United Nations Specialized Standards group whose membership includes common carriers concerned with devising and proposing recommendations for international telecommunications representing alphabets, graphics, control information, and other fundamental information interchange issues.

context handle. RPC: A reference to state (client context) maintained across remote procedure calls by a server on behalf of a client. See *client context*.

control access. CDS: An access right that grants users the ability to change the access control on a name and to perform other powerful management tasks, such as replicate a directory or move a clearinghouse.

convergence. CDS: The degree to which CDS attempts to keep all replicas of a directory consistent. Two factors control the persistence and speed at which CDS keeps directory replicas up to date: the setting of a directory's **CDS_Convergence** attribute (high, medium, or low) and the background skulk time. By default, every directory inherits the convergence setting of its parent.

copy. GDS, XDS: Either a copy of an entry stored in other DSAs through bilateral agreement or a locally and dynamically stored copy of an entry resulting from a request (a cache copy).

creation timestamp (CTS). An attribute of all CDS clearinghouses, directories, soft links, child pointers, and object entries that contains a unique value reflecting the date and time the name was created. The timestamp consists of two parts; a time portion and a portion containing the system identifier of the node on which the name was created. These two parts guarantee uniqueness among timestamps generated on different nodes.

credentials. Security: A general term for privilege attribute data that has been certified by a trusted privilege certification authority.

cross-linking information. In order for OS/390 DCE to provide RACF-DCE interoperability and single sign-on to DCE, DCE provides utilities (see **mvsexpt** and **mvsimpt**) to incorporate into RACF the information that associates an OS/390-RACF user ID with a DCE principal's identifying information and the DCE principal's UUID with the corresponding OS/390-RACF user ID. The information is placed in a RACF DCE segment and the RACF general resource class, DCEUUIDS. This is called **cross-linking information** and is what allows interoperability and single sign-on to work. See also *interoperability* and *single sign-on*.

CTS. Creation timestamp.

D

daemon. (1) A long-lived process that runs unattended to perform continuous or periodic system-wide functions such as network control Some daemons are triggered automatically to perform their task; others operate periodically. An example is the **cron** daemon, which periodically performs the tasks listed in the **crontab** file. Many standard dictionaries accept the spelling *demon*. (2) A DCE server process.

DCE. Distributed Computing Environment.

directory. (1) A logical unit for storing entries under one name (the directory name) in a CDS namespace. Each physical instance of a directory is called a replica. (2) A collection of open systems that cooperates to hold a logical database of information about a set of objects in the real world.

directory ID. Directory identifier.

Directory Service. A DCE component. The Directory Service is a central repository for information about resources in a distributed system. See *Cell Directory Service* and *Global Directory Service*.

distributed computing. A type of computing that allows computers with different hardware and software to be combined on a network, to function as a single computer, and to share the task of processing application programs.

Distributed Computing Environment (DCE). A comprehensive, integrated set of services that supports the development, use, and maintenance of distributed applications. DCE is independent of the operating system and network; it provides interoperability and portability across heterogeneous platforms.

Distributed File Service. A DCE component. Distributed File Service joins the local file systems of several file server machines making the files equally available to all Distributed File Service client machines. Distributed File Service allows users to access and share files stored on a file server anywhere in the network, without having to consider the physical location of the file. Files are part of a single, global name space, so that a user can be found anywhere in the network by means of the same name.

Distributed Time Service (DTS). A DCE component. It provides a way to synchronize the times on different hosts in a distributed system.

DNS. Domain Name System.

Domain Name System (DNS). A hierarchical scheme for giving meaningful names to hosts in a TCP/IP network.

domain name. A unique network name that is associated with a network's unique address.

DTS. Distributed Time Service.

DTS entity. DTS: The server or clerk software on a system.

DUA cache. GDS: The part of the DUA that stores information to optimize name lookups. Each cache contains copies of recently accessed object entries as well as information about DSAs in the directory.

E

element. RPC: Any of the bits of a bit string, the octets of an octet string, or the octets by means of which the characters of a character string are represented.

endpoint. RPC: An address of a specific server instance on a host.

entity. (1) CDS: Any manageable element through the CDS namespace. Manageable elements include directories, object entries, servers, replicas, and clerks. The CDS control program (CDSCP) commands are based on directives targeted for specific entities. (2) DTS: See *DTS entity*.

entity type. DTS: An identifier of an entity that determines whether it is a server or a clerk.

entry. GDS, XDS: The part of the DIB that contains information relating to a single directory object. Each entry consists of directory attributes.

ENV. environment variable

environment variable (ENV). A variable included in the current software environment that is available to any called program that requests it.

exception. (1) An abnormal condition such as an I/O error encountered in processing a data set or a file. (2) One of five types of errors that can occur during a floating-point exception. These are valid operation, overflow, underflow, division by zero, and inexact results. [OSF] (3) Contrast with *interrupt, signal*.

export. (1) RPC: To place the server binding information associated with an RPC interface or a list of object UUIDs or both into an entry in a name service database. (2) To provide access information for an RPC interface. Contrast with *unexport*.

F

foreign cell. A cell other than the one to which the local machine belongs. A foreign cell and its binding information are stored in either GDS or the Domain Name System (DNS). The act of contacting a foreign cell is called intercell. Contrast with *local cell*.

G

GDA. Global Directory Agent.

GDS. Global Directory Service.

Global Directory Agent (GDA). A DCE component that makes it possible for the local CDS to access names in foreign cells. The GDA provides a connection to foreign cells through either the GDS or the Domain Name System (DNS).

Global Directory Service (GDS). A DCE component. A distributed replicated directory service that provides a global namespace that connects the local DCE cells into one worldwide hierarchy. DCE users can look up a name outside a local cell with GDS.

global name. A name that is universally meaningful and usable from anywhere in the DCE naming environment. The prefix /... indicates that a name is global.

group. (1) RPC: A name service entry that corresponds to one or more RPC servers that offer common RPC interfaces, RPC objects, or both. A group contains the names of the server entries, other groups, or both that are members of the group. See *NSI group attribute*. (2) Security: Data that associates a named set of principals that can be granted common access rights. See *subject identifier*.

H

handle. RPC: An opaque reference to information. See *binding handle*, *context handle*, *interface handle*, *name service handle*, and *thread handle*.

high convergence. CDS: A setting that controls the degree to which CDS attempts to keep all replicas of a directory consistent. High convergence means CDS makes one attempt to immediately propagate an update to all replicas. If that attempt fails (for example, if one of the replicas is unavailable), the software schedules a

skulk for within one hour. Under normal circumstances, a skulk occurs at least once every twelve hours on a directory with high convergence. Setting a directory's **CDS_Convergence** attribute controls convergence. See *low convergence* and *medium convergence*.

home cell. Synonym for *local cell*.

host ID. Synonym for *network address*.

I

import. (1) RPC: To obtain binding information from a name service database about a server that offers a given RPC interface by calling the RPC NSI import operation. (2) RPC: To incorporate constant, type, and import declarations from one RPC interface definition into another RPC interface definition by means of the IDL import statement.

interface. RPC: A shared boundary between two or more functional units, defined by functional characteristics, signal characteristics, or other characteristics, as appropriate. The concept includes the specification of the connection of two devices having different functions. See *RPC interface*.

interface definition. RPC: A description of an RPC interface written in the DCE Interface Definition Language (IDL). See *RPC interface*.

interface handle. RPC: A reference in code to an interface specification. See *binding handle* and *interface specification*.

interface identifier. RPC: A string containing the interface Universal Unique Identifier (UUID) and major and minor version numbers of a given RPC interface. See *RPC interface*.

interface specification. RPC: An opaque data structure that is generated by the DCE IDL compiler from an interface definition. It contains identifying and descriptive information about an RPC interface. See *interface definition*, *interface handle*, and *RPC interface*.

interface UUID. RPC: The Universal Unique Identifier (UUID) generated for an RPC interface definition using the UUID generator. See *interface definition* and *RPC interface*.

International Organization for Standardization (ISO). An international body composed of the national standards organizations of 89 countries. ISO issues standards on a vast number of goods and services including networking software.

interoperability. The capability to communicate, execute programs, or transfer data among various functional units in a way that requires the user to have little or no knowledge of the unique characteristics of those units.

ISO. International Organization for Standardization

K

Kerberos. The authentication protocol used to carry out DCE private key authentication. Kerberos was developed at the Massachusetts Institute of Technology.

key. A value used to encrypt and decrypt data.

L

LAN. Local area network.

local. (1) Pertaining to a device directly connected to a system without the use of a communication line. (2) Pertaining to devices that have a direct, physical connection. Contrast with *remote*.

local area network (LAN). A network in which communication is limited to a moderate-sized geographical area (1 to 10 km) such as a single office building, warehouse, or campus, and which does not generally extend across public rights-of-way. A local network depends on a communication medium capable of moderate to high data rate (greater than 1Mbps), and normally operates with a consistently low error rate.

local cell. The cell to which the local machine belongs. Synonymous with *home cell*. Contrast with *foreign cell*.

local name. A name that is meaningful and usable only within the cell where an entry exists. The local name is a shortened form of a global name. Local names begin with the prefix /.: and do not contain a cell name. Synonymous with *cell-relative name*.

low convergence. A setting that controls the degree to which CDS attempts to keep all replicas of a directory consistent. Low convergence means CDS does not immediately propagate an update; it simply waits for the next skulk to distribute all updates that occurred since the last skulk. Skulks occur at least once every 24 hours on directories with low convergence. Low convergence helps conserve resources by avoiding update propagations between skulks. Setting a directory's **CDS_Convergence** attribute controls convergence. See *high convergence* and *medium convergence*.

M

manager. RPC: A set of remote procedures that implement the operations of an RPC interface and that can be dedicated to a given type of object. See also *object* and *RPC interface*.

mask. (1) A pattern of characters used to control the retention or deletion of portions of another pattern of characters (2) Security: Used to establish maximum permissions that can then be applied to individual ACL entries. (3) GDS: The administration screen interface menus.

master replica. CDS: The first instance of a specific directory in the namespace. After copies of the directory have been made, a different replica can be designated as the master, but only one master replica of a directory can exist at a time. CDS can create, update, and delete object entries and soft links in a master replica.

medium convergence. CDS: A setting that controls the degree to which CDS attempts to keep all replicas of a directory consistent. Medium convergence means CDS makes one attempt to immediately propagate an update to all replicas of the directory in which a change was made. If the attempt fails, the software lets the next scheduled skulk make the replicas consistent. Skulks occur at least once every 12 hours on a directory with medium convergence. When a name space is created, the default setting on the root directory is medium. Setting a directory's **CDS_Convergence** attribute controls convergence. See *high convergence* and *low convergence*.

mvsexpt. One of two (the other is **mvsimpt**) utilities used to automate much of the administrator's work in creating the cross-linking information for DCE-RACF interoperability. The **mvsexpt** utility creates the cross-linking information in the RACF database from information in the DCE registry. See also *cross-linking information*, *interoperability*, and *single sign-on*.

mvsimpt. One of two (the other is **mvsexpt**) utilities used to automate much of the administrator's work in creating the cross-linking information for DCE-RACF interoperability. The **mvsimpt** utility creates DCE principals from information obtained from the RACF database. See also *cross-linking information*, *interoperability*, and *single sign-on*.

N

name. GDS, CDS: A construct that singles out a particular (directory) object from all other objects. A name must be unambiguous (denote only one object); however, it need not be unique (be the only name that unambiguously denotes the object).

name service. A central repository of named resources in a distributed system. In DCE, this is the same as Directory Service.

name service handle. RPC: An opaque reference to the context used by the series of next operations called during a specific name service interface (NSI) search or inquiry.

name service interface (NSI). RPC: A part of the application program interface (API) of the RPC run time. NSI routines access a name service, such as CDS, for RPC applications.

namespace. CDS: A complete set of CDS names that one or more CDS servers look up, manage, and share. These names can include directories, object entries, and soft links.

network. A collection of data processing products connected by communications lines for exchanging information between stations.

network address. An address that identifies a specific host on a network. Synonymous with *host ID*.

Network Data Representation (NDR). RPC: The transfer syntax defined by the Network Computing Architecture. See *transfer syntax*.

network protocol. A communications protocol from the Network Layer of the Open Systems Interconnection (OSI) network architecture, such as the Internet Protocol (IP).

node. (1) An endpoint of a link, or a junction common to two or more links in a network. Nodes can be preprocessors, controllers, or workstations, and they can vary in routing and other functional capabilities. (2) In network topology, the point at an end of a branch. It is usually a physical machine.

NSI. Name service interface.

NSI binding attribute. RPC: An RPC-defined attribute (NSI attribute) of a name service entry; the binding attribute stores binding information for one or more interface identifiers offered by an RPC server and identifies the entry as an RPC server entry. See *binding information* and *NSI object attribute*. See also *server entry*.

NSI group attribute. RPC: An RPC-defined attribute (NSI attribute) of a name service entry that stores the entry names of the members of an RPC group and identifies the entry as an RPC group. See *group*.

NSI object attribute. RPC: An RPC-defined attribute (NSI attribute) of a name service entry that stores the object UUIDs of a set of RPC objects. See *object*.

NSI profile attribute. RPC: An RPC-defined attribute (NSI attribute) of a name service entry that stores a collection of RPC profile elements and identifies the entry as an RPC profile. See *profile*.

O

object. (1) A data structure that implements some feature and has an associated set of operations. (2) RPC: For RPC applications, anything that an RPC server defines and identifies to its clients using an object Universal Unique Identifier (UUID). An RPC object is often a physical computing resource such as a database, directory, device, or processor. Alternatively, an RPC object can be an abstraction that is meaningful to an application, such as a service or the location of a server. See *object UUID*. (3) XDS: Anything in the world of telecommunications and information processing that can be named and for which the directory information base (DIB) contains information. (4) XOM: Any of the complex information objects created, examined, changed, or destroyed by means of the interface.

object entry. CDS: The name of a resource (such as a node, disk, or application) and its associated attributes, as stored by CDS. CDS administrators, client application users, or the client applications themselves can give a resource an object name. CDS supplies some attribute information (such as a creation timestamp) to become part of the object, and the client application may supply more information for CDS to store as other attributes. See *entry*.

object name. CDS: A name for a network resource.

object UUID. RPC: The Universal Unique Identifier (UUID) that identifies a particular RPC object. A server specifies a distinct object UUID for each of its RPC objects. To access a particular RPC object, a client uses the object UUID to find the server that offers the object. See *object*.

Open Software Foundation (OSF). A nonprofit research and development organization set up to encourage the development of solutions that allow computers from different vendors to work together in a true open-system computing environment.

operation. (1) GDS: Processing performed within the directory to provide a service, such as a read operation. (2) RPC: The task performed by a routine or procedure that is requested by a remote procedure call.

organization. (1) The third field of a subject identifier. (2) Security: Data that associates a named set of users who can be granted common access rights that are usually associated with administrative policy.

OSF. Open Software Foundation.

P

package. XOM: A specified group of related object management (OM) classes, denoted by an object identifier.

parent directory. CDS: Any directory that has one or more levels of directories beneath it in a cell name space. A directory is the parent of any directory immediately beneath it in the hierarchy.

Partitioned data set (PDS). A data set in direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data.

password. A secret string of characters shared between a computer system and a user. The user must specify the character string to gain access to the system.

PDS. Partitioned data set

permission. (1) The modes of access to a protected object. The number and meaning of permissions with respect to an object are defined by the access control list (ACL) Manager of the object. (2) GDS: One of five groups that assigns modes of access to users: MODIFY PUBLIC, READ STANDARD, MODIFY STANDARD, READ SENSITIVE, or MODIFY SENSITIVE. Synonymous with *access right*. See also *access control list*.

person. See *principal*.

presentation address. An unambiguous name that is used to identify a set of presentation service access points. Loosely, it is the network address of an open systems interconnection (OSI) service.

primary name. The string name of an object to which any aliases for that object refer. The DCE refers to objects by their primary names, although DCE users may refer to them by their aliases.

principal. Security: An entity that can communicate securely with another entity. In the DCE, principals are represented as entries in the Registry database and include users, servers, computers, and authentication surrogates.

privilege attribute. Security: An attribute of a principal that may be associated with a set of permissions. DCE privilege attributes are identity-based and include the principal's name, group memberships, and local cell.

privilege ticket. Security: A ticket that contains the same information as a simple ticket, and also includes a

privilege attribute certificate. See *service ticket*, *simple ticket*, and *ticket-granting ticket*.

profile. RPC: An entry in a name service database that contains a collection of elements from which name service interface (NSI) search operations construct search paths for the database. Each search path is composed of one or more elements that refer to name service entries corresponding to a given RPC interface and, optionally, to an object. See *NSI profile attribute* and *profile element*.

profile element. RPC: A record in an RPC profile that maps an RPC interface identifier to a profile member (a server entry, group, or profile in a name service database). See *profile*. See also *group*, *interface identifier* and *server entry*.

protocol. A set of semantic and syntactic rules that determines the behavior of functional units in achieving communication.

protocol sequence. Synonym for *RPC protocol sequence*.

R

RACF. Resource Access Control Facility.

read-only replica. (1) CDS: A copy of a CDS directory in which applications cannot make changes. Although applications can look up information (read) from it, they cannot create, change, or delete entries in a read-only replica. Read-only replicas become consistent with other, changeable replicas of the same directory during skulks and routine propagation of updates. (2) Security: A replicated Registry server.

Registry database. Security: A database of security information about principals, groups, organizations, accounts, and security policies.

Registry Service. Security: One of three services provided by the Security Service; the Registry Service manages information about principals, accounts, and security policies. The other services are the Privilege Service and the Authentication Service.

remote. Pertaining to a device, file or system that is accessed by your system through a communications line. Contrast with *local*.

remote procedure. RPC: An application procedure located in a separate address space from calling code. See *remote procedure call*.

remote procedure call. RPC: A client request to a service provider located anywhere in the network.

Remote Procedure Call (RPC). A DCE component. It allows requests from a client program to access a procedure located anywhere in the network.

replica. CDS: A directory in the CDS namespace. The first instance of a directory in the name space is the master replica. See *master replica* and *read-only replica*.

request. A command sent to a server over a connection.

resource. Items such as printers, plotters, data storage, or computer services. Each has a unique identifier associated with it for naming purposes.

Resource Access Control Facility (RACF). An IBM licensed program, that provides for access control by identifying and verifying the users to the system, authorizing access to protected resources, and logging the detected unauthorized access to protected resources.

ROM. Read-only memory.

RPC. Remote Procedure Call.

RPC control program (RPCCP). An interactive administrative facility for managing name service entries and endpoint maps for RPC applications.

RPCCP. RPC control program

RPC interface. A logical group of operations, data types, and constant declarations that serves as a network contract for a client to request a procedure in a server. See also *interface definition* and *operation*.

RPC protocol. An RPC-specific communications protocol that supports the semantics of the DCE RPC API and runs over either connectionless or connection-oriented communications protocols.

RPC protocol sequence. A valid combination of communications protocols represented by a character string. Each RPC protocol sequence typically includes three protocols: a network protocol, a transport protocol, and an RPC protocol that works with the network and transport protocols. See *network protocol*, *RPC protocol*, and *transfer protocol*. Synonymous with *protocol sequence*.

S

Security Service. A DCE component that provides trustworthy identification of users, secure communications, and controlled access to resources in a distributed system.

segment. One or more contiguous elements of a string.

server. (1) On a network, the computer that contains programs, data, or provides the facilities that other computers on the network can access. (2) The party that receives remote procedure calls. Contrast with *client*.

server entry. RPC: A name service entry that stores the binding information associated with the RPC interfaces of a particular RPC server and object Universal Unique Identifiers (UUIDs) for any objects offered by the server. See also *binding information*, *NSI binding attribute*, *NSI object attribute*, *object* and *RPC interface*.

service. In network architecture, the capabilities that the layers closer to the physical media provide to the layers closer to the end user.

service ticket. Security: A ticket for a specified service other than the ticket-granting service. See *privilege ticket*, *simple ticket*, and *ticket-granting ticket*.

session. GDS: A sequence of directory operations requested by a particular user of a particular directory user agent (DUA) using the same session object management (OM) object.

shell script. A file containing shell commands. If the file can be processed, you can specify its name as a simple command. Processing of a shell script causes a shell to run the commands in the script. Alternatively, a shell can be requested to run the commands in a shell script by specifying the name of the shell script as the operand **sh** utility.

SID. Subject identifier.

signal. Threads: To wake only one thread waiting on a condition variable. See *broadcast*.

sign-on. (1) A procedure to be followed at a terminal or workstation to establish a link to a computer. (2) To begin a session at a workstation. (3) Same as log on or log in.

simple name. CDS: One element in a CDS full name. Simple names are separated by slashes in the full name.

simple ticket. Security: A ticket that contains the principal's identity, a session key, a timestamp and other information, sealed using the target's secret key. See *privilege ticket*, *service ticket*, and *ticket-granting ticket*.

single sign-on. In OS/390 DCE, single sign-on to DCE allows an OS/390 user who has already been authenticated to an MVS external security manager, such as RACF, to be logged in to DCE. DCE does this automatically when a DCE application is started, if the user is not already logged in to DCE.

soft link. CDS: A pointer that provides an alternative name for an object entry, directory, or other soft link in the name space. A soft link can be permanent or it can expire after a specific period of time. The CDS server also can delete it after the name that the link points to is deleted.

specific. XOM: The attribute types that can appear in an instance of a given class, but not in an instance of its superclasses.

standard. A model that is established and widely used.

string. An ordered sequence of bits, octets, or characters, accompanied by the string's length.

subject identifier (SID). A string that identifies a user or set of users. Each SID consists of three fields in the form person.group.organization. In an account, each field must have a specific value; in an access control list (ACL) entry, one or more fields may use a wildcard.

syntax. (1) XOM: An object management (OM) syntax is any of the various categories into which the OM specification statically groups values on the basis of their form. These categories are additional to the OM type of the value. (2) A category into which an attribute value is placed on the basis of its form. See *attribute syntax*.

T

thread handle. RPC: A data item that enables threads to share a storage management environment.

ticket. Security: An application-transparent mechanism that transmits the identity of an initiating principal to its target. See *privilege ticket*, *service ticket*, *simple ticket* and *ticket-granting ticket*.

ticket-granting ticket. Security: A ticket to the ticket-granting service. See *privilege ticket*, *service ticket*, and *simple ticket*.

transfer syntax. RPC: A set of encoding rules used for transmitting data over a network and for converting

application data to and from different local data representations. See also *Network Data Representation*.

type. XOM: A category into which attribute values are placed on the basis of their purpose. See *attribute type*.

type UUID. RPC: The Universal Unique Identifier (UUID) that identifies a particular type of object and an associated manager. See also *manager* and *object*.

U

unexport. RPC: To remove binding information from a server entry in a name service database. Contrast with *export*.

Universal Unique Identifier (UUID). RPC: An identifier that is immutable and unique across time and space. A UUID can uniquely identify an entity such as an object or an RPC interface. See *interface UUID*, *object UUID*, and *type UUID*.

update timestamp (UTS). CDS: An attribute that identifies the time at which the most recent change was made to any attribute of a particular CDS name. For directories, the UTS reflects changes made only to attributes that apply to the actual directory (not one of its replicas).

user. A person who requires the services of a computing system.

UTS. Update timestamp.

UUID. Universal unique identifier

V

value. XOM: An arbitrary and complex information item that can be viewed as a characteristic or property of an object. See *attribute value*.

W

workstation. A device that enables users to transmit information to or receive information from a computer, for example, a display station or printer.

X

X.500. The CCITT/ISO standard for the open systems interconnection (OSI) application-layer directory. It allows users to register, store, search, and retrieve information about any objects or resources in a network or distributed system.

Bibliography

This bibliography is a list of publications for OS/390 DCE and other products. The complete title, order number, and a brief description is given for each publication.

OS/390 DCE Publications

This section lists and provides a brief description of each publication in the OS/390 DCE library.

Overview

- *Distributed Computing Environment: Understanding the Concepts*, GC09-1478

  This book introduces Open Software Foundation (OSF) DCE. It describes the technology components of DCE, from a high-level overview to a discussion of the interdependencies among the components.

- *OS/390 DCE Introduction*, GC28-1581

  This book introduces OS/390 DCE. Whether you are a system manager, technical planner, OS/390 system programmer, or application programmer, it will help you understand DCE, and evaluate the uses and benefits of including OS/390 DCE as part of your information processing environment.

Planning

- *OS/390 DCE Planning*, SC28-1582

  This book helps you plan for the organization and installation of OS/390 DCE. It discusses the benefits of distributed computing in general, and describes how to develop plans for a distributed system in an OS/390 DCE environment.

Administration

- *OS/390 DCE Configuring and Getting Started*, SC28-1583

  This book helps system and network administrators configure OS/390 DCE.

- *OS/390 DCE Administration Guide*, SC28-1584

  This book helps system and network administrators understand OS/390 DCE, and tells how to administer it from the batch, TSO, and shell environments.

- *OS/390 DCE Command Reference*, SC28-1585

  This book provides reference information for the commands that system and network administrators use to work with OS/390 DCE.

- *OS/390 DCE User's Guide*, SC28-1586

  This book describes how to use OS/390 DCE to work with your user account, use the directory service, work with namespaces, and change access to objects that you own.

Application Development

- *OS/390 DCE Application Development Guide: Introduction and Style*, SC28-1587

  This book assists you in designing, writing, compiling, linking, and running distributed applications in OS/390 DCE.

- *OS/390 DCE Application Development Guide: Core Components*, SC28-1588

  This book assists programmers in developing applications using application facilities, threads, remote procedure calls, distributed time service, and security service.

- *OS/390 DCE Application Development Guide: Directory Services*, SC28-1589

  This book describes the OS/390 DCE directory service and assists programmers in developing applications for the cell directory service and the global directory service.

- *OS/390 DCE Application Development Reference*, SC28-1590

  This book explains the DCE Application Program Interfaces (APIs) that you can use to write distributed applications on OS/390 DCE.

- *OS/390 LDAP Client Application Development Guide and Reference*, SC24-5878

  This book describes the Lightweight Directory Access Protocol (LDAP) client APIs that you can use to write distributed applications on OS/390 DCE and gives you information on how to develop LDAP applications.

Reference

- *OS/390 DCE Messages and Codes*, SC28-1591

  This book provides detailed explanations and recovery actions for the messages, status codes, and exception codes issued by OS/390 DCE.

OS/390 Security Server Publications

This section lists and provides a brief description of books in the OS/390 Security Server library that may be needed for the OS/390 DCE Security Server and for RACF® interoperability.

- *OS/390 Security Server (DCE) Overview*, GC28-1938

  This book describes the DCE security server and provides a road map for DCE security server information in the OS/390 DCE library.

- *OS/390 Security Server (RACF) Security Administrator's Guide*, SC28-1915.

  This book explains RACF concepts and describes how to plan for and implement RACF.

- *OS/390 Security Server LDAP Server Administration and Usage Guide*, SC24-5861

  This book describes how to install, configure, and run the stand-alone LDAP daemon (SLAPD). It is intended for administrators who will maintain the server and database.

- *Firewall Technologies Guide and Reference*, SC24-5835

  This book provides the configuration, commands, messages, examples and problem determination for the OS/390 Firewall Technologies. It is intended for network or system security administrators who install, administer and use the OS/390 Firewall Technologies.

Tool Control Language Publication

- *Tcl and the Tk Toolkit*, John K. Osterhout, (c)1994, Addison—Wesley Publishing Company.

  This non-IBM book on the Tool Control Language is useful for application developers, DCECP script writers, and end users.

IBM C/C++ Language Publication

- *IBM OS/390 C/C++ Programming Guide*, SC09-2362

  This book describes how to develop applications in the C/C++ language in OS/390.

OS/390 DCE Application Support Publications

This section lists and provides a brief description of each publication in the OS/390 DCE Application Support library.

- *OS/390 DCE Application Support Configuration and Administration Guide*, SC24-5834

  This book helps system and network administrators understand and administer Application Support.

- *OS/390 DCE Application Support Programming Guide*, SC24-5833

  This book provides information on using Application Support to develop applications that can access CICS® and IMS™ transactions.

Encina Publications

- *OS/390 Encina Toolkit Executive Guide and Reference*, SC24-5832

  This book discusses writing Encina applications for OS/390.

- *OS/390 Encina Transactional RPC Support for IMS*, SC24-5874

  This book is to help software designers and programmers extend their IMS transaction applications to participate in a distributed, transactional client/server application.

Index

D

E

F

G

H

I

K

L

M

N

O

Readers' Comments

OS/390®
DCE
User's Guide

Publication No. SC28-1586-01

You may use this form to report errors, to suggest improvements, or to express your opinion on the appearance, organization, or completeness of this book.

Date: _____

IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Note

Report system problems to your IBM representative or the IBM branch office serving you.
U.S. customers can order publications by calling the IBM Software Manufacturing Solutions at **1-800-879-2755**.

In addition to using this postage-paid form, you may send your comments by:

FAX	1-607-752-2327	Internet	pubrcf@vnet.ibm.com
IBM Mail	USIB2L8Z@IBMMAIL	IBMLink	GDLVME(PUBRCF)

Would you like a reply? ___YES ___ NO If yes, please tell us the type of response you prefer.

___ Electronic address: _____

___ FAX number: _____

___ Mail: (Please fill in your name and address below.)

Name _____ Address _____

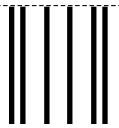
Company or Organization _____

Phone No. _____

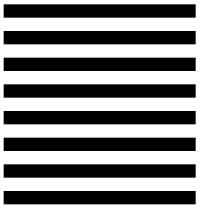
Readers' Comments
SC28-1586-01

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

Department G60
International Business Machines Corporation
Information Development
1701 North Street
ENDICOTT NY 13760-5553

Fold and Tape **Please do not staple** Fold and Tape

SC28-1586-01

Cut or Fold
Along Line

Program Number: 5647-A01

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC28-1586-01

Spine information:

OS/390 DCE User's Guide